# Guide to High Availability: Configuring boot/root/swap

*Jeannie Johnstone Kobert*

THE NETWORK IS THE COMPUTER™

**Sun Microsystems, Inc.**

901 San Antonio Road
Palo Alto, CA 94303-4900 USA
650 960-1300 Fax 650 969-9131

Part No.: 805-7711-05
July 1999, Revision 50

Copyright 1999 Sun Microsystems, Inc., 901 San Antonio Road • Palo Alto, CA 94303 USA. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, AnswerBook, Java, the Java Coffee Cup, Sun BluePrints and Solaris are trademarks, registered trademarks, or service marks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc. The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

**RESTRICTED RIGHTS**: Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g)(2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015(b)(6/95) and DFAR 227.7202-3(a).

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 1998 Sun Microsystems, Inc., 901 San Antonio Road • Palo Alto, CA 94303 Etats-Unis. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, AnswerBook, Java, le logo Jave Coffee Cup,Sun BluePrints, et Solaris sont des marques de fabrique ou des marques déposées, ou marques de service, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc. L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Please
Recycle

# Contents

**Acknowledgments xiii**

**Preface xv**

Sun BluePrints Program xv

About This Book xvi

Who Should Use This Book xvi

How This Book Is Organized xvii

Related Books xvii

Typographic Conventions xviii

Shell Prompts xviii

**1. Mirroring with the Sun StorEdge A5000 Subsystem and Solstice DiskSuite Software 1**

Hardware Configuration 1

Software Configuration 3

Creating Initial State Database Replicas 4

Two-Disk Configuration 5

▼ To Create Initial State Database Replicas 6

▼ To Mirror the Root Disk 13

Detecting and Recovering from a Disk Failure 20

▼ To Detect a Failed Component 20

▼ To Recover a Disk 24

**2. Mirroring with the Sun StorEdge A5000 Subsystem and the Sun Enterprise Volume Manager Software 35**

Configuring the SEVM Software 37

▼ To Configure the SEVM Software 37

Recovery 45

▼ To Recover the Primary Boot Disk 48

Adding Hot Relocation 56

**3. Sun StorEdge MultiPack or UniPack as a Solution 57**

Hardware Configuration 57

Software Configuration 58

▼ Command Line Configuration 60

Recovery 66

▼ To Recover the Failed Primary Boot Drive 67

Upgrading the Solaris Operating Environment 68

▼ To Unencapsulate the boot/root/swap Disk 69

**4. Alternative Boot Solutions 73**

Alternate Pathing and Sun Enterprise Volume Manager 73

Overview of Alternate Pathing 74

Hardware Configuration 74

Software Configuration 75

▼ To Configure Alternate Pathing 77

▼ To Mirror the Boot Disk 91

**A. Using the Command Line 95**

Setup 95

Recovery 98

# Code Samples


CODE EXAMPLE 1-1 `format` Command Used to Allocate Space for Each Database Replica 6

CODE EXAMPLE 1-2 `vfstab` File Showing Original Entries 13

CODE EXAMPLE 1-3 `vfstab` File Showing Updated Entries 16

CODE EXAMPLE 1-4 Final `vfstab` Reflecting All Metadevices 19

CODE EXAMPLE 1-5 Console Reports Error Message When Disk Fails 22

CODE EXAMPLE 1-6 `metastat` Command Reports Errors for Failed Mirrors 23

CODE EXAMPLE 1-7 `metadb` Command Reports Errors for Failed Replicas 24

CODE EXAMPLE 1-8 `ls` Command Showing Original Device Name Assignments 26

CODE EXAMPLE 1-9 `luxadm` Command to Remove Failed Old Primary Disk 26

CODE EXAMPLE 1-10 `luxadm` Command to Insert New Primary Disk 27

CODE EXAMPLE 1-11 `luxadm set_boot_dev` Command Used to Update the Records 28

CODE EXAMPLE 1-12 `ls` Command Showing Updated Device Assignments 28

CODE EXAMPLE 1-13 `luxadm` Command Sets Up the New Boot Drive 33

CODE EXAMPLE 2-1 `cat /etc/vx/cntrls.exclude` 36

CODE EXAMPLE 2-2 `cat /etc/vx/disks.exclude` 36

CODE EXAMPLE 2-3 Example of Volume Manager Path Name 37

CODE EXAMPLE 2-4 Encapsulating the Boot Disk 37

CODE EXAMPLE 2-5 Volume Manager Automatically Updates the `/etc/vfstab` File 40

CODE EXAMPLE 2-6 Mounted File Systems Shown with the `df` Command 40

CODE EXAMPLE 2-7 Lines Automatically Added to the `/etc/system` File 41

CODE EXAMPLE 2-8 `vxdiskadm` Command Used to Mirror Boot Disk 43

CODE EXAMPLE 2-9 `vxdisk list` Command 46

CODE EXAMPLE 2-10 `vxdisk list c4t1d0s2` Command 46

CODE EXAMPLE 2-11 `vxprint` Command Output 47

CODE EXAMPLE 2-12 `vxdiskadm` Command 48

CODE EXAMPLE 2-13 `vxdisk` Command 49

CODE EXAMPLE 2-14 `luxadm remove_device` Command 49

CODE EXAMPLE 2-15 `luxadm insert_device` Command 50

CODE EXAMPLE 2-16 `vxdiskadm` Option 5 - Replace a Failed or Removed Disk 51

CODE EXAMPLE 2-17 Unmirroring the Root Device 54

CODE EXAMPLE 2-18 Unencapsulating `root` 54

CODE EXAMPLE 2-19 Command-Line Tools Showing Results 55

CODE EXAMPLE 2-20 Results of `cat /etc/vfstab` Command 55

CODE EXAMPLE 2-21 Results of `mount` Command 56

CODE EXAMPLE 3-1 `vxdisk list` command 60

CODE EXAMPLE 3-2 Messages During Encapsulation Phase 60

CODE EXAMPLE 3-3 Output from the `/etc/vfstab` and `df` Commands 62

CODE EXAMPLE 3-4 `vxdisksetup` Command 63

CODE EXAMPLE 3-5 Adding Disks to `rootdg` 64

CODE EXAMPLE 3-6 `vxmirror` Command Used to Mirror the Root and Swap Disks 65

CODE EXAMPLE 3-7 Output Generated from the `vxdiskadm` Script 67

CODE EXAMPLE 3-8 The `vxunroot` Script 70

CODE EXAMPLE 3-9 Command-Line View of Disks Having Returned to Original State 70

CODE EXAMPLE 4-1 System Snapshot Before AP and SEVM 76

CODE EXAMPLE 4-2 List of Devices 76

CODE EXAMPLE 4-3 Volume Label Displayed with `prtvtoc` 77

CODE EXAMPLE 4-4 Errors Resulting From Not Allocating Space for AP Databases First 78

CODE EXAMPLE 4-5 AP Databases Displayed with `apconfig` 79

CODE EXAMPLE 4-6 `apinst` Used to View of List of Attached Controllers 81

CODE EXAMPLE 4-7 `apconfing -S -u` Command Used to View Uncommitted Entries 82

CODE EXAMPLE 4-8 Verifying the Database Contents 83

CODE EXAMPLE 4-9 `ls-ltr` Command Used to Verify the New Links 84

CODE EXAMPLE 4-10 Replacing the Physical Device Name With the Metadisk Name 84

CODE EXAMPLE 4-11 `/etc/vfstab` File Showing Metadisk Name 85

CODE EXAMPLE 4-12 AP `root` Information Found in the `/etc/system` File 85

CODE EXAMPLE 4-13 `cntrls.exclude` File, Forcing Only AP Metadevices to be Displayed 86

CODE EXAMPLE 4-14 `vxinstall` Command used to Install Volume Manager 87

# Figures


FIGURE 1-1 One Host/Server/Domain 2

FIGURE 1-2 Two Hosts/Servers/Domains 2

FIGURE 1-3 Four Hosts/Servers/Domains 3

FIGURE 1-4 Metastate Database Warning 9

FIGURE 1-5 Building a MetaDB 10

FIGURE 1-6 Disk View Window with View All Controllers Selected 10

FIGURE 1-7 Slice Browser Window Showing the Primary Root Disk 11

FIGURE 1-8 Uncommitted MetaDB 12

FIGURE 1-9 Completed MetaDB 12

FIGURE 1-10 Concat/Stripe Template 13

FIGURE 1-11 Concat/stripe Warning on a Mounted File System 14

FIGURE 1-12 Concat/Stripe Containing One Stripe 14

FIGURE 1-13 Concat/Stripe on a Mounted File System 15

FIGURE 1-14 One Way Concatenation of Root 15

FIGURE 1-15 Mirror Template 16

FIGURE 1-16 Unassigned, Uncommitted Mirror Object 17

FIGURE 1-17 Adding a Stripe Into a Mirror 17

FIGURE 1-18 Logging Provided by Metatool 18

FIGURE 1-19 Two-Way Mirror for Root Setup and Resynchronization 18

FIGURE 1-20 MetaDB Normal State 21

FIGURE 1-21 Two MetaDBs Lost 21

FIGURE 1-22 One of the Failed Slices Showing Status: Critical; Mirror Showing Status: Urgent: 22

FIGURE 1-23 Repairing the State Database Replicas 25

FIGURE 1-24 SDS Warning Message 25

FIGURE 1-25 MetaDB Contains Only the Two Slices from the Valid Secondary Disk 29

FIGURE 1-26 Slices from Newly Inserted Disk Added to the MetaDB 29

FIGURE 1-27 New Disk Slices Committed to the MetaDB 30

FIGURE 1-28 Primary Root Slice Reporting Critical State 30

FIGURE 1-29 Window Displayed by Double Clicking Failed Mirror Component 31

FIGURE 1-30 Component Marked Critical Ready to be Committed 31

FIGURE 1-31 Critical Component in Resynchronization State 32

FIGURE 2-1 GUI View of Encapsulated `rootdisk` 42

FIGURE 2-2 GUI View of Spare Disk for Mirroring Root 43

FIGURE 2-3 GUI View of the Mirrored `rootdisk` 45

FIGURE 2-4 Failure of Primary `boot` Disk 46

FIGURE 2-5 Disks Being Synchronized 53

FIGURE 2-6 The `rootdisk` Mirrored 53

FIGURE 2-7 The `rootdisk` Unmirrored 54

FIGURE 3-1 GUI View of Encapsulated Boot Disk 62

FIGURE 3-2 GUI View of new disks brought under SEVM control 64

FIGURE 3-3 GUI View of rootdg showing the results of the `addisk` command 65

FIGURE 3-4 GUI View of `vxmirror` 66

FIGURE 3-5 GUI View of `vxplex` 69

FIGURE 4-1 Sun StorEdge A5000 System with One I/O Controller Board Mirrored 75

FIGURE 4-2 GUI (VxVA) View of `rootdg` 91

FIGURE 4-3 Depiction of the Fully-Mirrored System Disk 93

FIGURE 4-4 System Snapshot Before AP and SEVM 93

# Acknowledgments

Thanks to my husband David Kobert, who believes that I can do anything. He quoted Goethe before my first interview with Sun Microsystems, “Whatever you can do, or dream you can, begin it. Boldness has genius, power and magic in it.” I believe him.

This book is dedicated to my friends and co-workers, with whom I have worked for the past ten years at the Fort Lauderdale site, which will regrettably close in the Fall of 1999. A very big thank you to Marcy Davis, Karen Bernstein, Joan Smith, and Susan Copeland for their friendship and support. Thanks to Keith Tripp and Judy Leach for the use of their equipment, and to Art Licht for his guidance and technical inspiration.

Thanks to Terrie Douglas for her technical expertise, input, and patience. Thanks to those who took the time to review and comment on this text: Kevin Siebenthal, Enrique Vargas, David Brean, Nivas Madhur, Jack Bochsler, Mario Ornelas, and Steve Heye.

Special thanks to Charles Alexander, whose managerial style provided inspiration, vision, and opportunity. Thanks to my group, Richard McDougall, Adrian Cockcroft, Barbara Jugo (Ms. BluePrint), Alice Kemp and Vibha Akkaraju, my technical writers, Enrique Vargas, Evert Hoogendoorn, Don Devitt, Tom Bialaski, Cathleen Plaziak, and David Deeths.

A final thank you to my father, George Johnstone. Dad is a retired mechanical engineer from whom I certainly inherited logic and perseverance. I must also acknowledge my mother, the late Emily Johnstone, the person whom I admire most. Even at a very young age, she represented to me what a powerful woman is. She became a registered nurse, while raising four children, as well as being involved in family, charity, religion, and community. In countless ways Mom and Dad have helped me to be who I am today.

# Preface

The concept of high availability can be applied to all aspects of a computer system. This book applies that concept to the system disk, which includes boot, root, and swap, using a subset of the hardware and all the software solutions that Sun Microsystems™ offers today. Although this book provides detailed examples, using specific pieces of hardware and software, the high-availability information can be extended to other parts of the computer product line. For example, this book specifically covers the Sun StorEdge™ A5000 subsystem, Sun StorEdge UniPack and MultiPack external desktop storage systems, Sun Enterprise 4000 and 5000 servers, and the Sun Ultra™ 2 workstation. However, the concepts in this book are applicable to the Sun Enterprise 3000 to 6000 server line, as well as the Sun Enterprise 250 and 450 servers (which make use of multiple internal disks), and all Sun StorEdge subsystems that fit the criteria of hot-swappability. These concepts are *not* applicable to subsystems whose sole feature is Raid 5 functionality. If a piece of equipment has the same characteristics as the systems in this book (for example hot-swappablility), you can apply to it the concepts discussed in this book.

## Sun BluePrints Program

The mission of Sun BluePrints™ Program is to empower Sun customers with the technical knowledge required to implement reliable, available, extensible, and secure information systems within the data center using Sun products. Sun BluePrints Program is managed by the Enterprise Engineering Group. This group provides a framework to identify, develop, and distribute best practices information that applies across the Sun product line. Technical subject matter experts in various areas contribute to the program and focus on the scope and usefulness of the information.

The Enterprise Engineering Group is the primary provider of the technical content of Sun BluePrints Program that includes books, guides, and online articles. Through these vehicles, Sun can provide guidance, installation and implementation experiences, real-life scenarios, and late-breaking technical information.

The bimonthly electronic magazine, Sun BluePrints OnLine, is located on the Web at `http://www.sun.com/blueprints`. To be notified about updates to Sun BluePrints Program, please register yourself on this site.

# About This Book

*Guide to High Availability: Configuring boot/root/swap* provides configuration and recovery information for Sun servers. This book provides a cookbook approach for a variety of solutions that ensure system disk availability. Each chapter includes both hardware and software high availability solutions.

The philosophy of this book is that it is possible to achieve 100 percent uptime for the system boot disk. If the system disk is properly configured and its disk is hot-swappable, the server does not have to be rebooted when the system disk fails. This book tries to adhere strictly to the definition of high availability that allows a system to keep operating even with a failure.

Although this book does not cover every product and every configuration in detail, it provides enough information so that you can extrapolate the procedures to create the best solution for your particular server or disk.

# Who Should Use This Book

This book is intended for system engineers responsible for configuring a Sun server with a high availability requirement for the system disk. It uses an explicit, step-by-step approach, allowing you to have all the required information in one document. It covers a specific area of hardware and software configuration; for in-depth information on a particular component, consult the user manual for that product.

# How This Book Is Organized

**Chapter 1 “Mirroring with the Sun StorEdge A5000 Subsystem and Solstice DiskSuite Software”** is a recipe for mirroring your system disk using Solstice DiskSuite™ software and the Sun StorEdge A5000 system. It contains a specific example of configuration and one recovery scenario.

**Chapter 2 “Mirroring with the Sun StorEdge A5000 Subsystem and the Sun Enterprise Volume Manager Software”** is a recipe for mirroring your system disk using Sun Enterprise Volume Manager™ and the Sun StorEdge A5000 system. It contains a specific configuration example and one recovery example. This recovery example is different from the example presented in Chapter 1.

**Chapter 3 “Sun StorEdge MultiPack or UniPack as a Solution”** is a recipe for mirroring your system disk using the Sun StorEdge MultiPack or UniPack.

**Chapter 4 “Alternative Boot Solutions”** provides the steps that enable you to configure your boot disk to use alternate pathing in conjunction with Sun Enterprise Volume Manager software.

**Appendix A “Using the Command Line”** provides command-line options that correspond to the GUI steps used in each chapter.

# Related Books

For more information on related topics, see the following books:

- *Sun Enterprise Volume Manager 2.5 System Administrator’s Guide* (PN 805-1607-10)
- *Solstice DiskSuite 4.1 User’s Guide* (PN 802-4215)
- *Sun Enterprise Server Alternate Pathing 2.1 User’s Guide* (PN 805-3532-10)
- `luxadm` Manual Page

# Typographic Conventions

**TABLE P-1** Typographic Conventions

| Typeface | Meaning | Examples |
|---|---|---|
| `AaBbCc123` | The names of commands, files, and directories; on-screen computer output | Edit your `.login` file.<br>Use `ls -a` to list all files.<br>`% You have mail.` |
| **`AaBbCc123`** | What you type, when contrasted with on-screen computer output | `% su`<br>`Password:` |
| *AaBbCc123* | Book titles, new words or terms, words to be emphasized | Read Chapter 6 in the *User's Guide.*<br>These are called *class* options.<br>You *must* be superuser to do this. |
| | Command-line variable; replace with a real name or value | To delete a file, type `rm` *filename.* |

# Shell Prompts

**TABLE P-2** Shell Prompts

| Shell | Prompt |
|---|---|
| C shell | *machine_name*% |
| C shell superuser | *machine_name*# |
| Bourne shell and Korn shell | $ |
| Bourne shell and Korn shell superuser | # |

CHAPTER 1

# Mirroring with the Sun StorEdge A5000 Subsystem and Solstice DiskSuite Software

With the system disk as the focal point, this chapter defines the best solutions for configuring boot, root, and swap for a single domain or server. The solutions range from the smallest to the largest configuration. This chapter provides a cookbook approach to configuring the Sun StorEdge A5000 subsystem (hereafter referred to as the A5000 subsystem) as a system disk using Solstice DiskSuite (SDS) software. The focus of these configurations is on high availability, fault tolerance, robustness, simplicity, and flexibility of the system disk. In all configurations, the A5000 subsystem provides an external system disk, which avoids all electromagnetic interference (EMI) issues within a server's cabinet. The following text spells out the most desirable configurations, hardware and software components, and possible risk factors, if any exist, for the system disk. All configurations presented in this BluePrint have been verified.

## Hardware Configuration

The following configurations focus on high availability and failover capabilities for the system disk. This configuration section shows one, two, and four host/domain scenarios, easily scaling to eight hosts/domains.

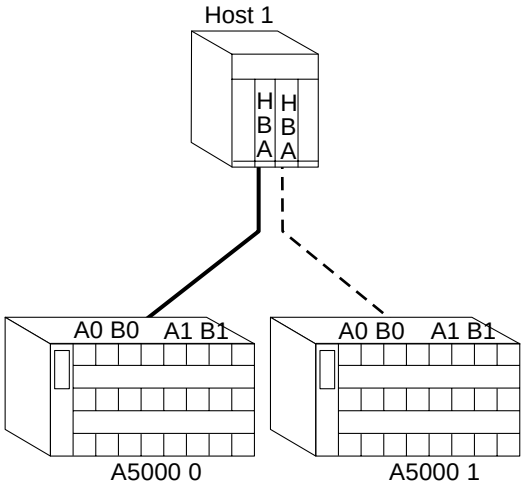


**FIGURE 1-1** One Host / Server / Domain

In FIGURE 1-1, two A5000 subsystems are used to enable 100 percent access to the system disk in the event of a power failure to one of the A5000 subsystems. The primary boot drive (solid line) is attached to A5000 subsystem 0, through port zero of Interface Board A. In case of a hardware failure, Host 1 would continue to access the mirrored boot drive attached to A5000 subsystem 1. Also note that in FIGURE 1-1, two host bus adapters are used in case of an adapter failure.

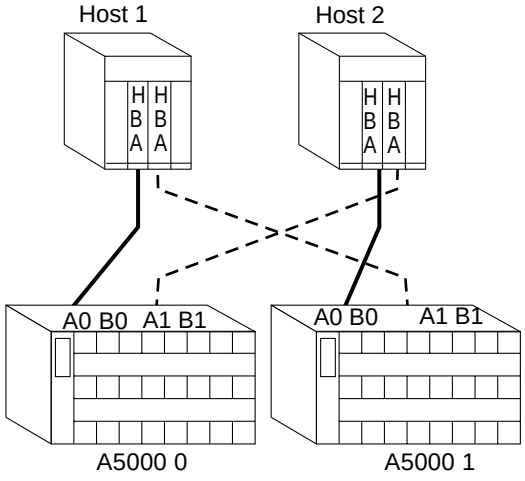


**FIGURE 1-2** Two Hosts / Servers / Domains

FIGURE 1-2 shows a split-loop configuration. Each system disk (two primaries and two secondaries), exists on its own loop. This configuration simply adds another server to the configuration in FIGURE 1-1. If there is a power failure on one Sun StorEdge A5000 subsystem, each host will access the system boot drive from the same A5000 subsystem. However, they would separate the drive's separate loops. This avoids all possible corruption from another host. Host 2 cannot access Host 1's primary drive located on loop 0.

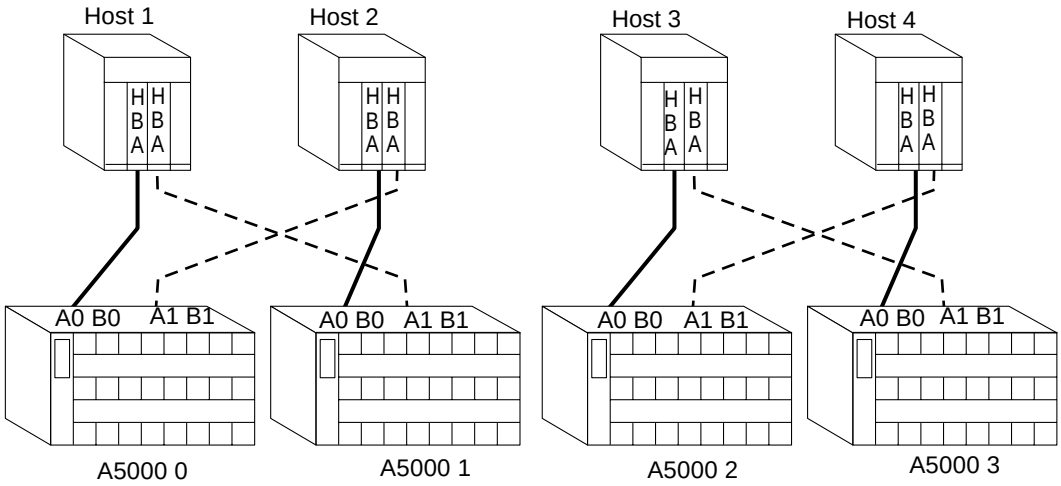


**FIGURE 1-3** Four Hosts / Servers / Domains

FIGURE 1-3 is simply a doubling of resources of the configuration in FIGURE 1-2, with the same high availability precautions and fault tolerant properties. No additional resources are shared. In an eight-domain configuration, this figure would again be doubled. Note that this configuration shows a host or domain as having dedicated drives on two Sun StorEdge A5000 subsystems. Host 1 can view disk drives from Sun StorEdge A5000 subsystem 0, loop 0, and Sun StorEdge A5000 subsystem 1, loop 1. This would presently total 14 disks. To prevent system disk corruption, hosts cannot access each other's disks. The use of hubs in such a configuration is not covered in this book.

# Software Configuration

To have the system disk drive available 100 percent of the time, it is necessary to mirror root, `swap`, and `/usr`. The Solstice DiskSuite software can mirror these file systems because the state database locations are patched into the metadisk driver via

the `/etc/system` file. The kernel, therefore, has knowledge of the state database early in the boot process. It can obtain all the information it needs about all metadevices. The instructions in this chapter use the Graphical User Interface (GUI). See Appendix A for the command-line method.

Solstice DiskSuite software is currently located on the Solaris™ Easy Access Server 2.0 Software CD, the successor to the Solaris Server Intranet Extension 1.0 suite of products. The instructions in this chapter were verified using Solstice DiskSuite 4.2; the package list included SUNWabmd (*Solstice DiskSuite 4.2 Administration Guide AnswerBook*), SUNWmd (Solstice DiskSuite), SUNWmdg (Solstice DiskSuite Tool), and SUNWmdn. No patches were necessary at the writing of this document. I installed Solstice DiskSuite 4.2 using the `pkgadd` command. Refer to the *Solstice DiskSuite 4.2 Administration Guide* for installation and setup instructions.

It is important to read the Solstice DiskSuite 4.2 documentation suite in conjunction with this BluePrint. This document is intended to provide a thorough guide to configuring your boot/root/swap for high availability, but it does not stand alone as a complete Solstice DiskSuite manual.

---

**Note –** The steps in the following procedures include clicking on an object within metatool and dragging it to its destination. Clicking on the object with the left mouse button, and using the middle mouse button to move the object greatly reduces GUI frustration. If an object refuses to be dragged onto the canvas, double click it to place it onto the canvas.

---

# Creating Initial State Database Replicas

A state database replica stores SDS configuration and state information. Before you can use the SDS software, you must create state database replicas. This section describes how to create initial state database replicas using different system configurations. The sample scenarios depict the user starting from scratch, with no database replicas defined.

The configuration for the database replicas depends on the availability of disks and how the slices on those disks are configured. If you do not have available slices on your system, you will not be able to use the SDS software to mirror your system disk. The configuration defined requires two disks, one on each Sun StorEdge A5000 subsystem. This configuration will result in 100 percent availability for a single point of failure to the system disk.

**Note –** In the case of a double failure, one to the system disk (that is, disk failure), and one to the system (system panic), a manual intervention to reboot from the secondary disk is required due to the majority consensus algorithm under the SDS software.

## Two-Disk Configuration

Depending on how the slices on your system disks are configured, you have three choices. The most desirable configuration, which is provided as an example, is the availability of two free slices on each of your primary and secondary system disks. Place one database replica on each of the available slices, for a total of four replicas. If your system disk has one free slice, you can either create two database replicas on one slice, or create the database replicas on a slice that will be part of a simple metadevice. Remember that a state database replica can not be part of root (`/`), `swap`, or `/usr`, an existing file system, or a slice containing data. A database replica can also exist within space specifically reserved for a replica within a striped or concatenated metadevice or a logging device. If you consider this option, be aware of the restrictions on using existing file systems and databases, and consider the need for additional disks on which to place the replicas.

Starting with a two-disk configuration, define two initial state database replicas (MetaDBs) on each boot disk on separate slices. This gives you the most protection for your two-disk system. Preferred recovery procedures, which follow, include recovery from a single point of failure to the system. These procedures will not require a reboot except in the event of a system panic.

Each replica occupies 517 Kbytes, or 1034 disk blocks of the partition. In the case of a larger disk, you will have to allocate at least one sector. For example, on a 9 Gbyte disk, one cylinder would be equal to 3591 blocks or approximately 1.75 Mbytes, which is greater than the requirement. The space must be allocated before initializing the state database replicas. You can choose the `format`, or `fmthard` command, or Disk Manager to accomplish this.

**Note –** In the event of a double failure, for example loss of the primary system disk in conjunction with a system panic, rebooting from the secondary system disk, although possible, would require administrator intervention. Because SDS software uses the majority consensus algorithm, you must have 51 percent of the total replicas available for the system to reboot.

## ▼ To Create Initial State Database Replicas

Use three windows in DiskSuite Tool to identify and view your configuration, and then put state database replicas on slices.

1. **As `root`, use the `format` command to allocate at least 517Kbytes, or 1034 sectors, for each state database replica you plan to create.**

   In this example, four partitions will be allocated. Each is 1.75 Mbyte or 3591 sectors in size. They reside on two system disks.

**CODE EXAMPLE 1-1** `format` Command Used to Allocate Space for Each Database Replica

```
# format
Searching for disks...done


AVAILABLE DISK SELECTIONS:
       0. c0t0d0 <SUN2.1G cyl 2733 alt 2 hd 19 sec 80>
          /sbus@1f,0/SUNW,fas@e,8800000/sd@0,0
       1. c1t2d0 <SUN4.2G cyl 3880 alt 2 hd 16 sec 135>
          /sbus@1f,0/SUNW,fas@0,8800000/sd@2,0
       2. c1t3d0 <SUN4.2G cyl 3880 alt 2 hd 16 sec 135>
          /sbus@1f,0/SUNW,fas@0,8800000/sd@3,0
       3. c1t4d0 <SUN4.2G cyl 3880 alt 2 hd 16 sec 135>
          /sbus@1f,0/SUNW,fas@0,8800000/sd@4,0
       4. c1t5d0 <SUN4.2G cyl 3880 alt 2 hd 16 sec 135>
          /sbus@1f,0/SUNW,fas@0,8800000/sd@5,0
       5. c1t8d0 <SUN4.2G cyl 3880 alt 2 hd 16 sec 135>
          /sbus@1f,0/SUNW,fas@0,8800000/sd@8,0
       6. c1t9d0 <SUN4.2G cyl 3880 alt 2 hd 16 sec 135>
          /sbus@1f,0/SUNW,fas@0,8800000/sd@9,0
       7. c2t0d0 <SUN9.0G cyl 4924 alt 2 hd 27 sec 133>
          /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370ca48f,0
       8. c2t4d0 <SUN9.0G cyl 4924 alt 2 hd 27 sec 133>
          /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370ca049,0
       9. c2t19d0 <SUN9.0G cyl 4924 alt 2 hd 27 sec 133>
          /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370c5993,0
      10. c2t21d0 <SUN9.0G cyl 4924 alt 2 hd 27 sec 133>
          /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370ca02a,0
- hit space for more or s to select -
Specify disk (enter its number)[7]: 7
selecting c2t0d0
[disk formatted]
format> part


PARTITION MENU:
        0      - change ‘0’ partition
```

**CODE EXAMPLE 1-1** `format` Command Used to Allocate Space for Each Database Replica

```
        1      - change ‘1’ partition
        2      - change ‘2’ partition
        3      - change ‘3’ partition
        4      - change ‘4’ partition
        5      - change ‘5’ partition
        6      - change ‘6’ partition
        7      - change ‘7’ partition
        select - select a predefined table
        modify - modify a predefined partition table
        name   - name the current table
        print  - display the current table
        label  - write partition map and label to the disk
        !<cmd> - execute <cmd>, then return
        quit
partition> print
Current partition table (unnamed):
Total disk cylinders available: 4924 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0 - 1140        1.95GB    (1141/0/0)  4097331
  1       swap    wu    1141 - 2281        1.95GB    (1141/0/0)  4097331
  2     backup    wm       0 - 4923        8.43GB    (4924/0/0) 17682084
  3 unassigned    wm       0               0         (0/0/0)           0
  4 unassigned    wm       0               0         (0/0/0)           0
  5 unassigned    wm    2282 - 3137        1.47GB    (856/0/0)   3073896
  6        usr    wm    3138 - 3993        1.47GB    (856/0/0)   3073896
  7 unassigned    wm       0               0         (0/0/0)           0

partition> 3
Part      Tag    Flag     Cylinders        Size            Blocks
  3 unassigned    wm       0               0         (0/0/0)           0

Enter partition id tag[unassigned]: alternates
Enter partition permission flags[wm]:
Enter new starting cyl[0]: 3994
Enter partition size[0b, 0c, 0.00mb, 0.00gb]: 1c
partition> print
Current partition table (unnamed):
Total disk cylinders available: 4924 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0 - 1140        1.95GB    (1141/0/0)  4097331
  1       swap    wu    1141 - 2281        1.95GB    (1141/0/0)  4097331
  2     backup    wm       0 - 4923        8.43GB    (4924/0/0) 17682084
  3 alternates    wm    3994 - 3994        1.75MB    (1/0/0)        3591
  4 unassigned    wm       0               0         (0/0/0)           0
  5 unassigned    wm    2282 - 3137        1.47GB    (856/0/0)   3073896
```

**CODE EXAMPLE 1-1** `format` Command Used to Allocate Space for Each Database Replica

```
  6        usr    wm    3138 - 3993        1.47GB    (856/0/0)   3073896
  7 unassigned    wm       0               0         (0/0/0)
m    2282 - 3137        1.47GB    (856/0/0)   3073896
  6        usr    wm    3138 - 3993        1.47GB    (856/0/0)   3073896
  7 unassigned    wm       0               0         (0/0/0)           0

partition> 4
Part      Tag    Flag     Cylinders        Size            Blocks
  4 unassigned    wm       0               0         (0/0/0)           0

Enter partition id tag[unassigned]: alternates
Enter partition permission flags[wm]:
Enter new starting cyl[0]: 3995
Enter partition size[0b, 0c, 0.00mb, 0.00gb]: 1c
partition> print
Current partition table (unnamed):
Total disk cylinders available: 4924 + 2 (reserved cylinders)

Part      Tag    Flag     Cylinders        Size            Blocks
  0       root    wm       0 - 1140        1.95GB    (1141/0/0)  4097331
  1       swap    wu    1141 - 2281        1.95GB    (1141/0/0)  4097331
  2     backup    wm       0 - 4923        8.43GB    (4924/0/0) 17682084
  3 alternates    wm    3994 - 3994        1.75MB    (1/0/0)        3591
  4 alternates    wm    3995 - 3995        1.75MB    (1/0/0)        3591
  5 unassigned    wm    2282 - 3137        1.47GB    (856/0/0)   3073896
  6        usr    wm    3138 - 3993        1.47GB    (856/0/0)   3073896
  7 unassigned    wm       0               0         (0/0/0)           0

partition> label
Ready to label disk, continue? y

partition> quit


FORMAT MENU:
        disk       - select a disk
        type       - select (define) a disk type
        partition  - select (define) a partition table
        current    - describe the current disk
        format     - format and analyze the disk
        repair     - repair a defective sector
        label      - write label to the disk
        analyze    - surface analysis
        defect     - defect list management
        backup     - search for backup labels
        verify     - read and display labels
        save       - save new disk/partition definitions
```

**CODE EXAMPLE 1-1** `format` Command Used to Allocate Space for Each Database Replica

```
        inquiry    - show vendor, product and revision
        volname    - set 8-character volume name
        !<cmd>     - execute <cmd>, then return
        quit
format> quit
#
```

2. **As `root`, start DiskSuite Tool to identify and view your configuration:**

```
# metatool &
```

3. **The first time you run DiskSuite Tool, a message appears stating that no database replicas exist. Click OK to continue.**

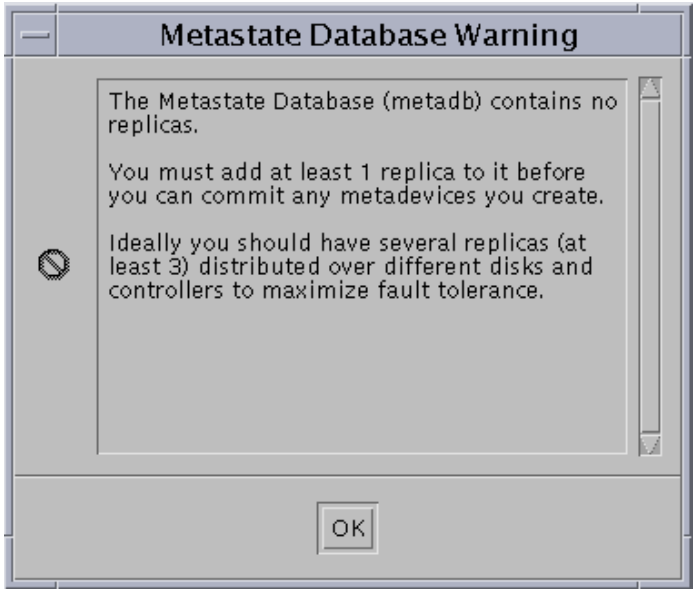
Metastate Database Warning
The Metastate Database (metadb) contains no replicas.
You must add at least 1 replica to it before you can commit any metadevices you create.
Ideally you should have several replicas (at least 3) distributed over different disks and controllers to maximize fault tolerance.
OK

**FIGURE 1-4** Metastate Database Warning

4. **From the Objects list, drag the MetaDB object onto the canvas.**

   If no MetaDBs are defined, the status is Critical, as shown in FIGURE 1-5.

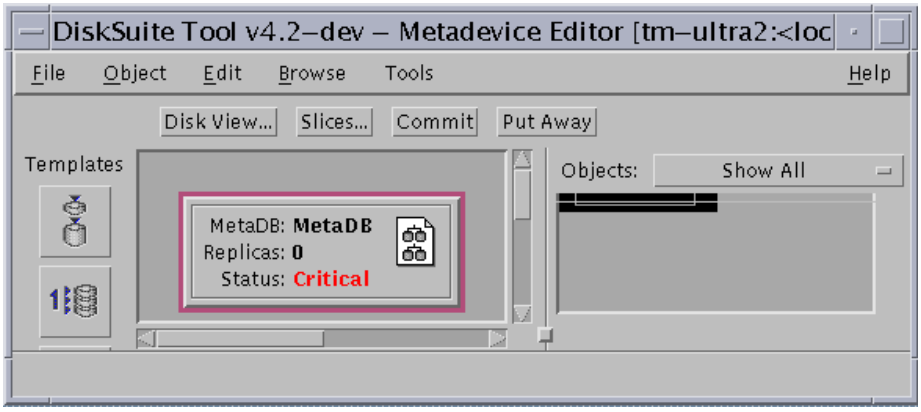


**FIGURE 1-5** Building a MetaDB

**5. Click Disk View to display the Disk View window.**

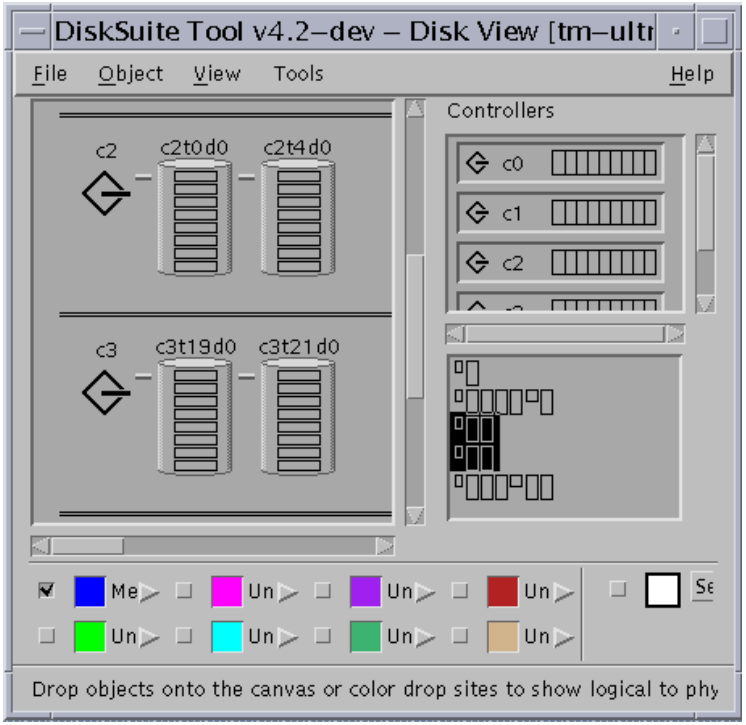


**FIGURE 1-6** Disk View Window with View All Controllers Selected

6. **Select View All Controllers from the View menu.**

   This shows your storage configuration: controllers, disks, and slices. You may want to use the View menu to set the view to 50, 100, or 200 percent viewing. The default is 100 percent. FIGURE 1-6 displays the disks we have chosen to store our MetaDBs for our primary and secondary mirrors.

7. **Click Slices to display the Slice Browser window.**

   This window shows the status and current use of slices. (The Disk View window's status line also displays this information when you position the cursor over the slice.)

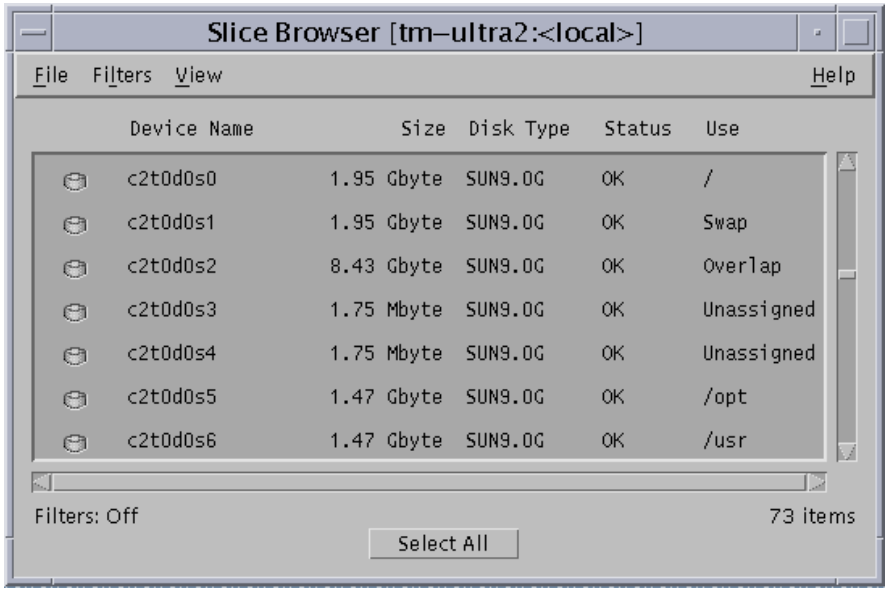


**FIGURE 1-7** Slice Browser Window Showing the Primary Root Disk

8. **Build a MetaDB object (create the state database replicas) by dragging available slices from the disks shown in the Disk View window or from the Slice Browser window over the MetaDB object shown in** FIGURE 1-8**.**

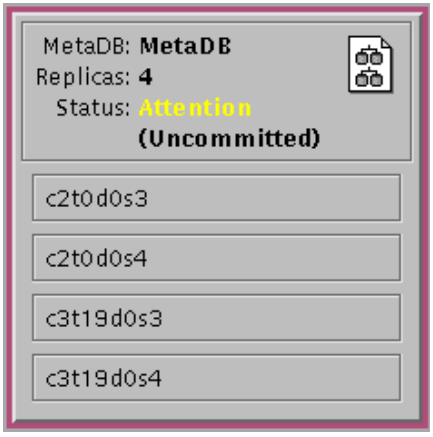


**FIGURE 1-8** Uncommitted MetaDB

Remember to always commit those changes, if you want to affect the system's current configuration. The result will be similar to FIGURE 1-9.

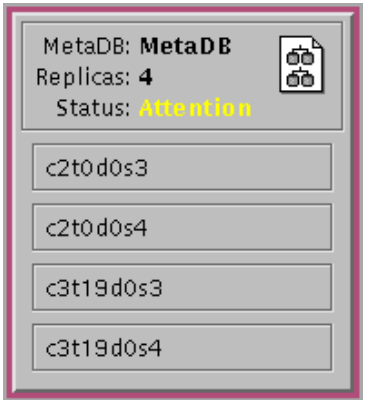


**FIGURE 1-9** Completed MetaDB

FIGURE 1-9 shows a committed MetaDB object consisting of four slices, each containing one state database replica. Notice that the target numbers are representative of the Sun StorEdge A5000 naming convention. The status of the MetaDB object is "Attention," indicating that the state database replicas are spread across two controllers.

Try to balance state database replicas across controllers to achieve redundancy. For configurations with many disks, put a state database replica on each disk. (You can also click the Set Filters button in the Disk View window to filter for slices that are not in use.)

## ▼ To Mirror the Root Disk

Before proceeding, make sure that you have partitioned a disk that is identical to the root disk you are about to mirror. If you have not already done so, exit metatool now and use `format`, `fmthard`, or Disk Manager to repartition your target secondary disk. You must exit metatool because immediate changes to any disk will not be reflected until metatool is stopped and restarted and the disk configuration is re-read. If possible, use disks consisting of the same disk geometries. Historically, UFS used disk blocks based on disk geometries, and today a mirror composed of disks with different geometries will only be as fast as its slowest disk. Although you can use a slice of equal or greater size than the slice you are mirroring (to avoid wasting space), it is more desirable to use slices of the same size when creating submirrors.

CODE EXAMPLE 1-2 shows the `vfstab` file before any modifications are made.

**CODE EXAMPLE 1-2** `vfstab` File Showing Original Entries

```
#cat /etc/vfstab
#device             device              mount     FS      fsck    mount      mount
#to mount           to fsck             point     type    pass    at boot    options
#
#/dev/dsk/c1d0s2    /dev/rdsk/c1d0s2    /usr      ufs     1       yes        -
fd                  -                   /dev/fd   fd      -       no         -
/proc               -                   /proc     proc    -       no         -
/dev/dsk/c2t0d0s1   -                   -         swap    -       no         -
/dev/dsk/c2t0d0s0   /dev/rdsk/c2t0d0s0  /         ufs     1       no         -
/dev/dsk/c2t0d0s6   /dev/rdsk/c2t0d0s6  /usr      ufs     1       no         -
/dev/dsk/c2t0d0s5   /dev/rdsk/c2t0d0s5  /opt      ufs     2       yes        -
swap                -                   /tmp      tmpfs   -       yes        -
```

1. **Set up the mirror for the root slice. Click the Concat/Stripe template, shown in** FIGURE 1-10**.**

**FIGURE 1-10** Concat/Stripe Template

DiskSuite Tool displays an unassigned and uncommitted Concat/Strip object on the canvas and gives it a metadevice name.

2. **Click Slices to display the Slice Browser window, and drag the slice containing the file system you want to mirror (slice 0, root) to the top rectangle of the Concat/Stripe object.** FIGURE 1-12 **displays the result.**

   The Slice Browser menu will list this with a device name equal to c2t0d0s0, and a component name of (/). You will see the warning window shown in FIGURE 1-11; choose Continue.

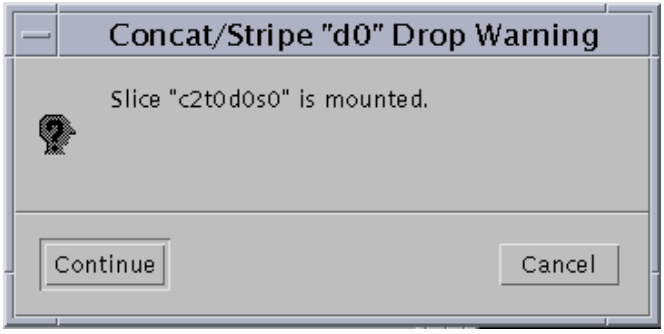


**FIGURE 1-11** Concat/stripe Warning on a Mounted File System

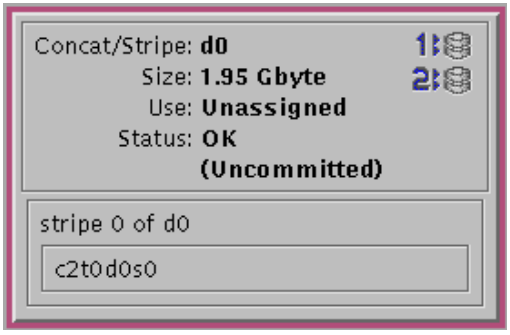


**FIGURE 1-12** Concat/Stripe Containing One Stripe

3. **Click the top rectangle of the Concat/Stripe object, then click Commit. When you see the Commit Warning shown in** FIGURE 1-13**, choose Really Commit and continue.**

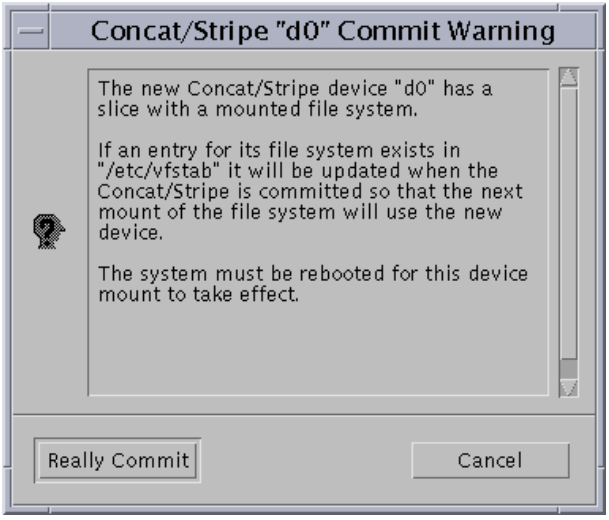
Concat/Stripe "d0" Commit Warning
The new Concat/Stripe device "d0" has a slice with a mounted file system.
If an entry for its file system exists in "/etc/vfstab" it will be updated when the Concat/Stripe is committed so that the next mount of the file system will use the new device.
The system must be rebooted for this device mount to take effect.
Really Commit
Cancel

**FIGURE 1-13** Concat/Stripe on a Mounted File System

This creates a one-way concatenation containing the file system, which will be used as the first submirror.

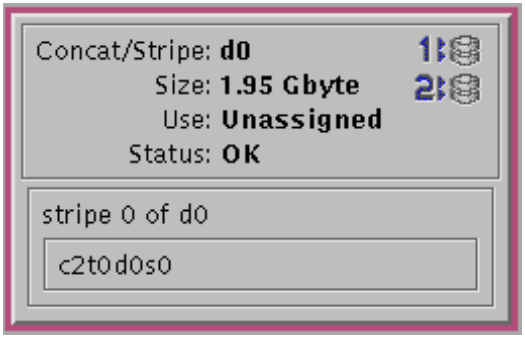


**FIGURE 1-14** One Way Concatenation of Root

The `/etc/vfstab` file will be automatically updated to use the concatenation's metadevice name, as shown in CODE EXAMPLE 1-3.

**CODE EXAMPLE 1-3** `vfstab` File Showing Updated Entries

```
#cat /etc/vfstab
#device             device               mount     FS      fsck    mount      mount
#to mount           to fsck              point     type    pass    at boot    options
#
#/dev/dsk/c1d0s2    /dev/rdsk/c1d0s2     /usr      ufs     1       yes        -
fd                  -                    /dev/fd   fd      -       no         -
/proc               -                    /proc     proc    -       no         -
/dev/dsk/c2t0d0s1   -                    -         swap    -       no         -
/dev/md/dsk/d0      /dev/md/rdsk/d0      /         ufs     1       no         -
/dev/dsk/c2t0d0s6   /dev/rdsk/c2t0d0s6   /usr      ufs     1       no         -
/dev/dsk/c2t0d0s5   /dev/rdsk/c2t0d0s5   /opt      ufs     2       yes        -
swap                -                    /tmp      tmpfs   -       yes        -
```

Do the same with the intended mirror slice of the secondary boot device, as described in the following steps:

4. **Click on the Concat/Stripe template.**
5. **Drag the (secondary root, device name c3t19d0s0, Use equals Unassigned) slice from the Slice Browser Window to this Concat/Stripe object.**
6. **Click the top rectangle of the Concat/Stripe object; then click Commit.**

   This creates the second submirror.
7. **Click the mirror template, shown in** FIGURE 1-15**.**

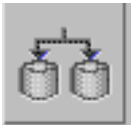

**FIGURE 1-15** Mirror Template

DiskSuite Tool displays an unassigned and uncommitted Mirror object on the canvas and gives it a metadevice name, as shown in FIGURE 1-16.

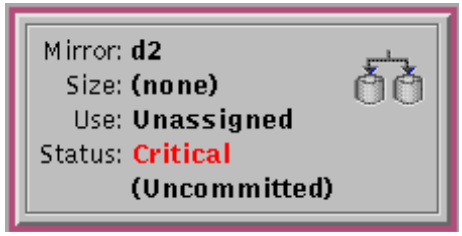


**FIGURE 1-16** Unassigned, Uncommitted Mirror Object

8. **Drag the Concat/Stripe object (`c2t0d0s0`, Concat/Stripe `d0`) onto the new Mirror object.**

   FIGURE 1-17 displays the unassigned, uncommitted mirror containing the Concat/Stripe object. The `/etc/vfstab` file is automatically updated to use the mirror's metadevice name.

**FIGURE 1-17** Adding a Stripe Into a Mirror

9. **Commit this change by clicking the top rectangle of the Mirror object; then click Commit.**

10. **Repeat Steps 1 through 9 for `swap`, `/usr`, or any other unmountable file system, and reboot the system.**

11. **Restart DiskSuite Tool to finish creating the two-way mirrors.**

    You are reminded to attach the submirrors you have set up but not yet attached, as shown in FIGURE 1-18.

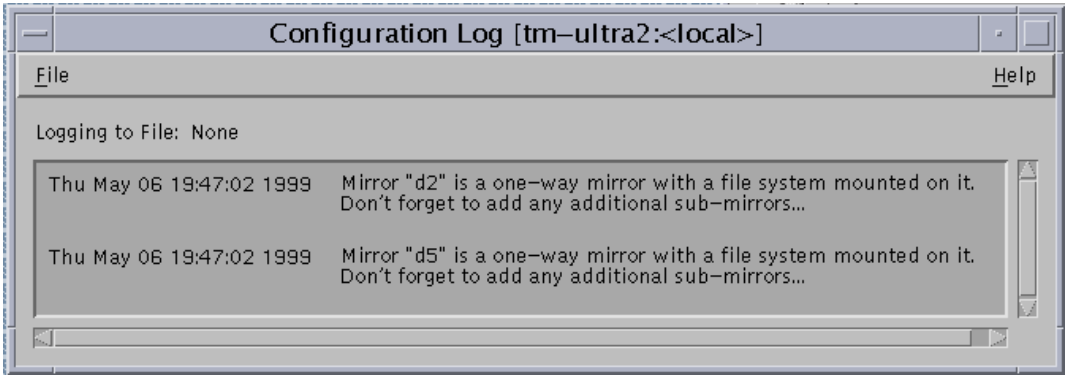
Configuration Log [tm–ultra2:<local>]
File
Help
Logging to File: None
Thu May 06 19:47:02 1999
Mirror "d2" is a one–way mirror with a file system mounted on it.
Don't forget to add any additional sub–mirrors...
Thu May 06 19:47:02 1999
Mirror "d5" is a one–way mirror with a file system mounted on it.
Don't forget to add any additional sub–mirrors...

**FIGURE 1-18** Logging Provided by Metatool

12. **Attach/Drag the second Concat/Stripe object to the Mirror object to create the two-way mirror.**

13. **Click the mirror and then the Commit button to commit the mirror.**

    Data from the first submirror is resynchronized to the second mirror. FIGURE 1-19 shows the assigned, committed mirror while it is resynchronizing.

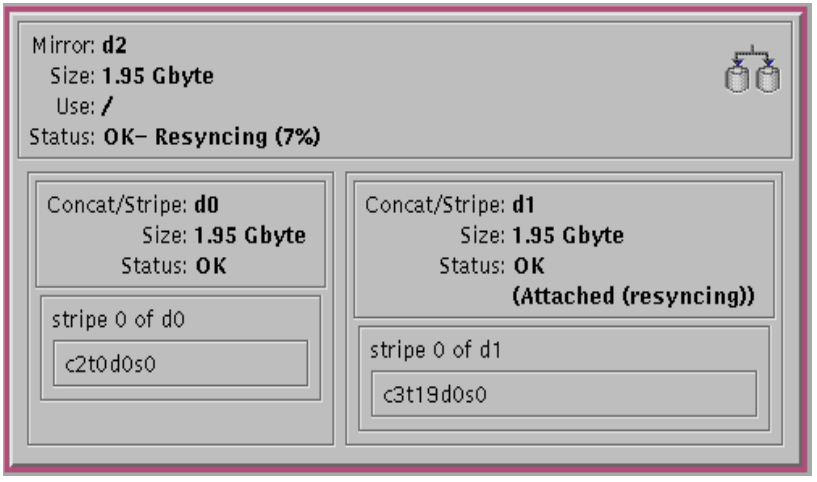


**FIGURE 1-19** Two-Way Mirror for Root Setup and Resynchronization

You can now mirror any other mountable devices (for example `/export/home`) on the boot disk,. The difference in this setup is that `metainit` will not run and that no reboot is necessary. An example is not provided.

**CODE EXAMPLE 1-4** Final `vfstab` Reflecting All Metadevices

```
#cat /etc/vfstab
#device             device              mount      FS       fsck    mount      mount
#to mount           to fsck             point      type     pass    at boot    options
#
#/dev/dsk/c1d0s2    /dev/rdsk/c1d0s2    /usr       ufs      1       yes        -
fd                  -                   /dev/fd    fd       -       no         -
/proc               -                   /proc      proc     -       no         -
/dev/md/dsk/d5      -                   -          swap     -       no         -
/dev/md/dsk/d2      /dev/md/rdsk/d2     /          ufs      1       no         -
/dev/md/dsk/d11     /dev/md/rdsk/d11    /usr       ufs      1       no         -
/dev/md/dsk/d8      /dev/md/rdsk/d8     /opt       ufs      2       yes        -
swap                -                   /tmp       tmpfs    -       yes        -
```

14. **Record the alternate boot device path in a location other than your root file system. You can determine the path to the alternate root device by using the `ls -l` command on the slice that is being attached as the second submirror to the root (/) mirror.**

```
#ls -l /dev/rdsk/c3t19d0s0
lrwxrwxrwx   1 root     root          74 Mar 23 10:33 /dev/rdsk/
c3t19d0s0 -> ../../devices/sbus@1f,0/SUNW,socal@1,0/sf@1,0/
ssd@w21000020370c5993,0:a,raw
```

With OpenBoot™ PROM, you can use the OpenBoot `nvalias` command to define a “backup root” `devalias` for the secondary root mirror:

```
ok nvalias backup_root /sbus@1f,0/SUNW,socal@1,0/sf@1,0/
ssd@w21000020370c5993,0:a
```

If a primary root disk failure occurs, boot from the secondary disk:

```
ok boot backup_root
```

---

**Note –** The Sun StorEdge A5000 subsystem is a hot-swapable storage solution. If the primary disk fails, Solstice DiskSuite software will continue to run from the secondary mirrored drive. This allows you to replace the failed primary drive without rebooting the system.

---

15. **If additional disks and controllers are available, define three MetaDBs across three disks, spread across power boundaries.**

    This configuration is necessary if the system requires a clean reboot, without operator intervention, onto a secondary device, which would be the case in the event of a double failure. Clean reboot is possible since more than 51 percent of the state database replicas would be available. Of course, in a 100 percent available configuration, a reboot would not be desirable.

# Detecting and Recovering from a Disk Failure

Mirroring all the system disks allows you to easily and quickly recover from possible Sun StorEdge A5000 subsystem hardware failures. Recovery from the following list of failures was verified:

- Fibre cable
- Gigabit Interface Converter (GBIC)
- Interface board
- Disk

This section guides you through the recovery process.

The system will continue to run on the mirrored disk that has not had a hardware interruption. When you are made aware of the hardware failure, you can hot-swap the failed component and bring the affected system disk back to a steady state. There are a number of ways to detect the failed component. One way is to use the `luxadm` command, which can indicate whether an interface board, GBIC, disk, or connection is no longer valid.

Once the defective device is swapped out, the mirror that is out of sync can be resynchronized. Order is important in this procedure, so follow the instructions closely. The steps in this chapter guide you through an actual system disk loss and the replacement of that system disk. Everything on the system disk is mirrored.

FIGURE 1-20 shows the system starting in a normal state of operation.

## ▼ To Detect a Failed Component

Look though the following possibilities to detect a failed component. Note that the graphics here represent one error; they do not cover every possible error.

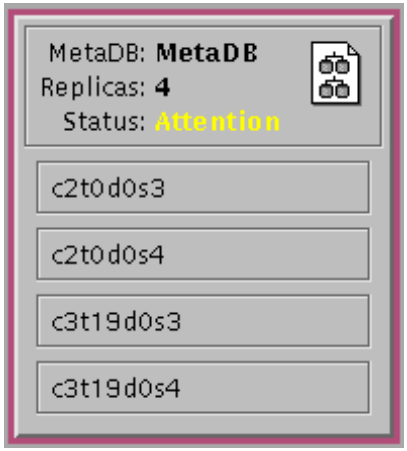


**FIGURE 1-20** MetaDB Normal State

In FIGURE 1-20, the primary system disk has not yet been lost. Errors shown on the console appear in the following pages.

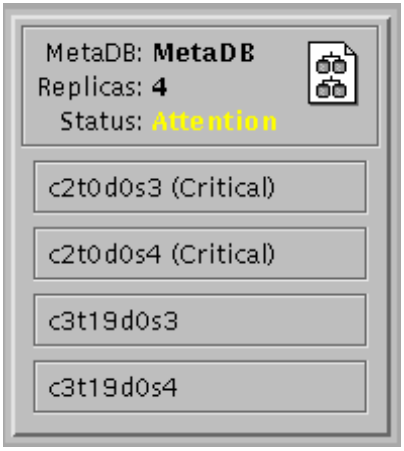


**FIGURE 1-21** Two MetaDBs Lost

Note the changes in the MetaDB GUI in FIGURE 1-21, when the error does occur. In addition, the root slice is now in a critical state, as shown in FIGURE 1-22. Access to the mirrored root slice is through the secondary drive, which is on another loop. The name of that disk is `c3t19d0s0`.

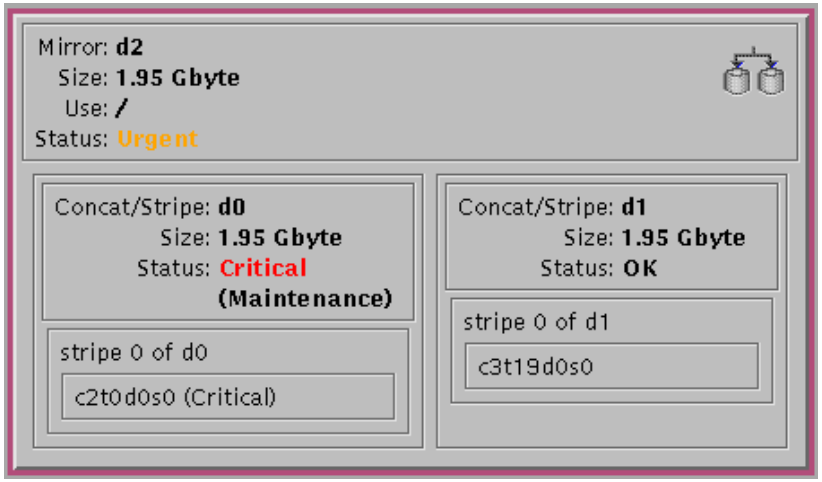


**FIGURE 1-22** One of the Failed Slices Showing Status: Critical; Mirror Showing Status: Urgent:

**CODE EXAMPLE 1-5** Console Reports Error Message When Disk Fails

```
WARNING: /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370c0c1c,0 (ssd2):
        requeue of command fails (fffffffe)

WARNING: /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370c0c1c,0 (ssd2):
        requeue of command fails (fffffffe)

WARNING: /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370c0c1c,0 (ssd2):
        transport rejected (-2)

WARNING: /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370c0c1c,0 (ssd2):
        transport rejected (-2)

WARNING: /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370c0c1c,0 (ssd2):
        transport rejected (-2)

WARNING: md: d0: write error on /dev/dsk/c2t0d0s0
WARNING: md: d0: /dev/dsk/c2t0d0s0 needs maintenance
WARNING: /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370c0c1c,0 (ssd2):
        transport rejected (-2)

WARNING: md: d9: write error on /dev/dsk/c2t0d0s6
WARNING: md: d9: /dev/dsk/c2t0d0s6 needs maintenance
```

**CODE EXAMPLE 1-6** `metastat` Command Reports Errors for Failed Mirrors

```
# metastat
d2: Mirror
    Submirror 0: d0
      State: Needs maintenance
    Submirror 1: d1
      State: Okay
    Pass: 1
    Read option: roundrobin (default)
    Write option: parallel (default)
    Size: 4097331 blocks
d0: Submirror of d2
    State: Needs maintenance
    Invoke: metareplace d2 c2t0d0s0 <new device>
    Size: 4097331 blocks
    Stripe 0:
       Device            Start Block  Dbase State       Hot Spare
        c2t0d0s0                   0     No    Maintenance
d1: Submirror of d2
    State: Okay
    Size: 4097331 blocks
    Stripe 0:
       Device            Start Block  Dbase State       Hot Spare
        c3t19d0s0                  0     No    Okay
```

The errors shown in CODE EXAMPLE 1-6 are displayed for all failed mirrors.

Notice in CODE EXAMPLE 1-7, under the flags column, that the failed replicas are no longer active and are reporting write errors.

**CODE EXAMPLE 1-7** `metadb` Command Reports Errors for Failed Replicas

```
# metadb -i
        flags           first blk       block count
     Wm  p  l        16            1034          /dev/dsk/c2t0d0s3
     W   p  l        16            1034          /dev/dsk/c2t0d0s4
    a    p  luo      16            1034         /dev/dsk/c3t19d0s3
    a    p  luo      16            1034         /dev/dsk/c3t19d0s4
 o - replica active prior to last mddb configuration change
 u - replica is up to date
 l - locator for this replica was read successfully
 c - replica's location was in /etc/opt/SUNWmd/mddb.cf
 p - replica's location was patched in kernel
 m - replica is master, this is replica selected as input
 W - replica has device write errors
 a - replica is active, commits are occurring to this replica
 M - replica had problem with master blocks
 D - replica had problem with data blocks
 F - replica had format problems
 S - replica is too small to hold current data base
 R - replica had device read errors
```

To create these failures, a loaded system was run and important system files were modified. The system was also forced to swap while the primary boot disk failed, and the secondary system disk kept the system running.

When you are informed that the system disk has shut down, either through a Simple Network Management Protocol (SNMP) trap issued through the Sun Enterprise SyMON™ software, or possibly a phone-home situation, you can hot-swap the primary system disk. You will be required to force the system to recognize the new disk, with the `luxadm` command, and then bring the new system disk up to date by resynchronizing it with the secondary disk.

## ▼ To Recover a Disk

**Note –** Before proceeding with the disk replacement and resynchronizing the failed mirrors, you must delete the MetaDBs. If you replace the failed disk first, and then try to replace the MetaDBs, SDS software will not recognize the new disk.

1. **Remove the invalid MetaDBs.**

   Remove the slices by simply dragging them out or by clicking the right mouse button and choosing remove. The result is shown in FIGURE 1-23.

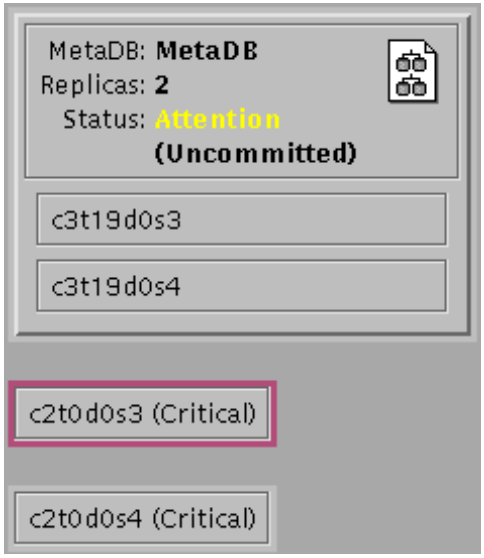


**FIGURE 1-23** Repairing the State Database Replicas

2. **Choose Commit to commit changes.**

   During the repair, replicas exist on one controller, so you will see the warning shown in FIGURE 1-24. Proceed through the warning message by clicking Really Commit.

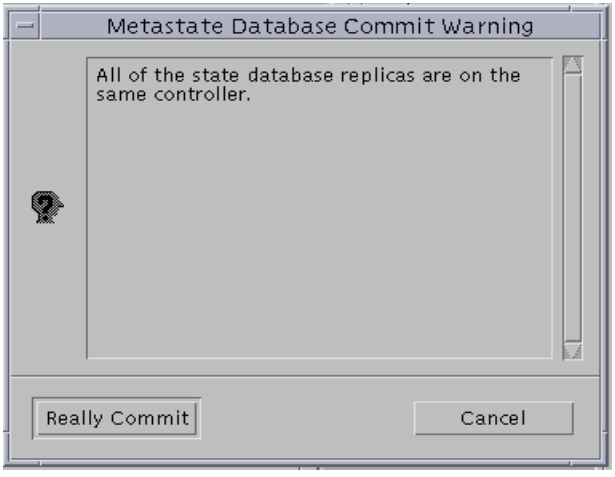


**FIGURE 1-24** SDS Warning Message

An `ls` command will show the old world wide number (WWN), as shown in CODE EXAMPLE 1-8. We are about to update the WWN with the following `luxadm` commands. In this example, the old WWN was 22000020370c0c1c,0:a, and its logical device is c0t1d0s0; the logical number will remain the same while the WWN will change.

**CODE EXAMPLE 1-8** `ls` Command Showing Original Device Name Assignments

```
# ls -ltr /dev/dsk/c2t0d0s0
lrwxrwxrwx   1 root     other         75 Apr 26 18:05 /dev/rdsk/
c2t0d0s0 -> ../../devices/sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370c0c1c,0:a
```

3. **Exit metatool.**

   You need to exit metatool in order to run `luxadm`. Otherwise, SDS will not recognize disk changes.

4. **Replace the primary system disk or the failed system disk.**

   Use the `luxadm remove_device` command to remove the logical devices tied to the old WWN, as shown in CODE EXAMPLE 1-9.

**CODE EXAMPLE 1-9** `luxadm` Command to Remove Failed Old Primary Disk

```
# luxadm remove_device A5000-1,f0

  WARNING!!! Please ensure that no filesystems are mounted on these
device(s).
  All data on these devices should have been backed up.

The list of devices which will be removed is:
  1: Box Name "A5000-1" front slot 0

Please enter 'q' to Quit or <Return> to Continue:

Notice: Drive in "A5000-1" front slot 0 may already be removed.

Hit <Return> after removing the device(s).

  Drive in Box Name "A5000-1" front slot 0
```

CODE EXAMPLE 1-9 reports that the device may have already been removed. (The primary disk was physically removed to simulate the loss of the primary disk.) We can now use the `luxadm insert_command` to put the new primary system disk into the A5000 subsystem.

**CODE EXAMPLE 1-10** `luxadm` Command to Insert New Primary Disk

```
# luxadm insert_device A5000-1,f0

The list of devices which will be inserted is:
  1: Box Name "A5000-1" front slot 0

Please enter 'q' to Quit or <Return> to Continue:

Hit <Return> after inserting the device(s).
WARNING: /sbus@1f,0/SUNW,socal@1,0/sf@0,0 (sf0):
        wwn changed on target 0x0

  Drive in Box Name "A5000-1" front slot 0
  Logical Nodes under /dev/dsk and /dev/rdsk :
        c2t0d0s0
        c2t0d0s1
        c2t0d0s2
        c2t0d0s3
        c2t0d0s4
        c2t0d0s5
        c2t0d0s6
        c2t0d0s7
```

CODE EXAMPLE 1-10 shows how and when the WWN and its associated devices are updated.

5. **Once you replace the primary system disk, the WWN is changed. Update your records to reflect that change, as shown in** CODE EXAMPLE 1-11**.**

   For example, if the machine has maintenance performed on it, the OpenBoot boot-device parameter should be updated to reflect the above change so that the machine can reboot.

**CODE EXAMPLE 1-11** `luxadm set_boot_dev` Command Used to Update the Records

```
# luxadm set_boot_dev /devices/sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370ca48f,0:a
Do you want to change boot-device to the new setting? (y/n) y
# eeprom boot-device
boot-device=/sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370ca48f,0:a

--or--

telnet> send break
Type  'go' to resume
{1} ok setenv boot-device /sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370ca48f,0:a
boot-device =         /sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370ca48f,0:a
{1} ok go
```

CODE EXAMPLE 1-12 shows the updated device assignments.

**CODE EXAMPLE 1-12** `ls` Command Showing Updated Device Assignments

```
# ls -ltr /dev/dsk/c2t0d0s0
lrwxrwxrwx   1 root     other         75 May  7 10:56 /dev/rdsk/
c2t0d0s0 -> ../../devices/sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370ca48f,0:a
```

Notice in CODE EXAMPLE 1-12 that the device c2t0d0s0 now is associated with 22000020370ca48f,0:a. The new disk is now recognized.

6. **Use** `format`**,** `fmthard`**, or Disk Manager to partition this new disk to be identical to the disk you have just replaced.**

   It can also be partitioned to be identical to the current online mirror.

7. **To resynchronize the new primary system disk, restart metatool.**

8. **Add the new disk's slices to the existing MetaDBs.**

   Select the disk and its slices from Slices, and drag them onto the MetaDB, shown in FIGURE 1-26.

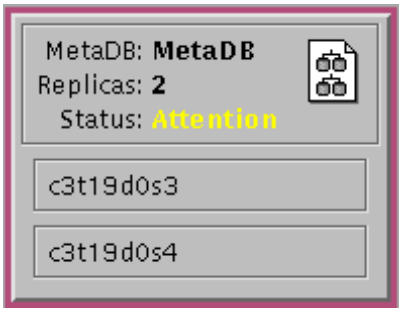


**FIGURE 1-25** MetaDB Contains Only the Two Slices from the Valid Secondary Disk

**FIGURE 1-26** Slices from Newly Inserted Disk Added to the MetaDB

FIGURE 1-27 shows the MetaDB state after the new slices are committed.

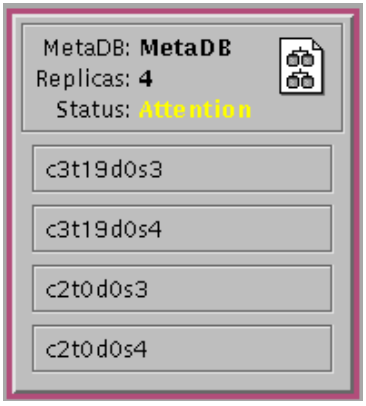


**FIGURE 1-27** New Disk Slices Committed to the MetaDB

You have successfully replaced the two failed MetaDBs.

9. **Proceed with fixing all the mirrors. Start with / (root), slice c2t0d0s0.**

   To start resynchronizing the metadevice that is marked with a red Critical state on each mirror of your primary boot disk, double-click the failed component itself. This action displays another GUI window containing slice information, as shown in FIGURE 1-29.

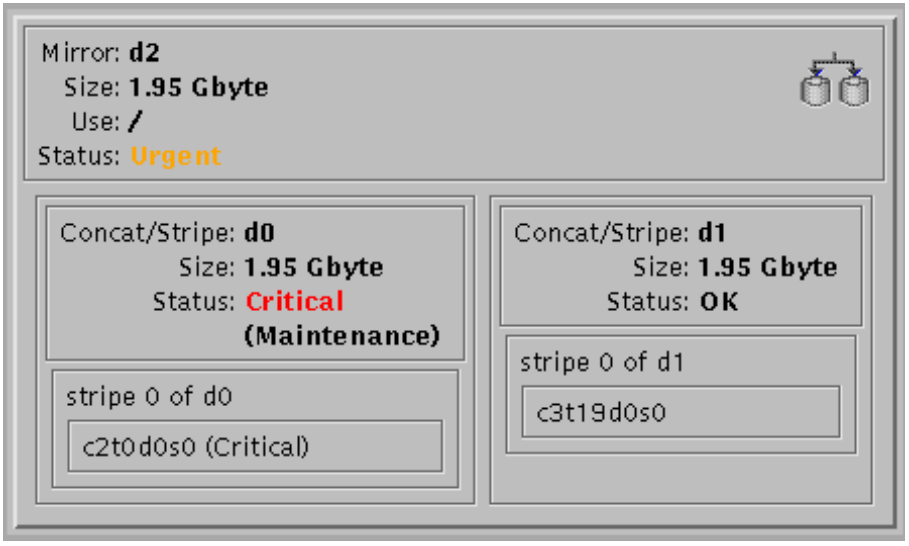


**FIGURE 1-28** Primary Root Slice Reporting Critical State

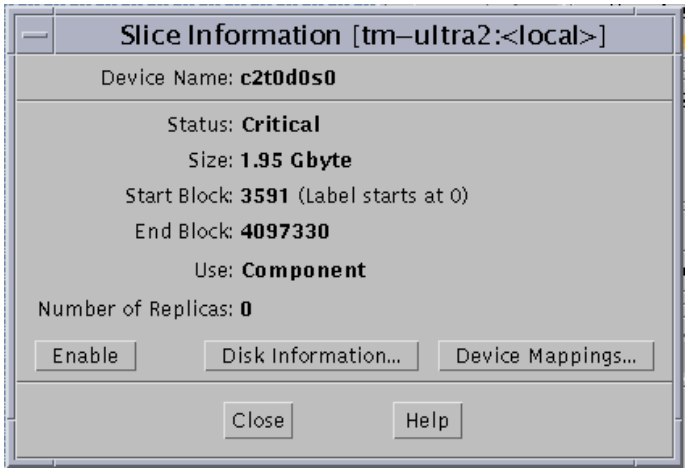


**FIGURE 1-29** Window Displayed by Double Clicking Failed Mirror Component

10. **Click the Enable button, and then commit that choice.**

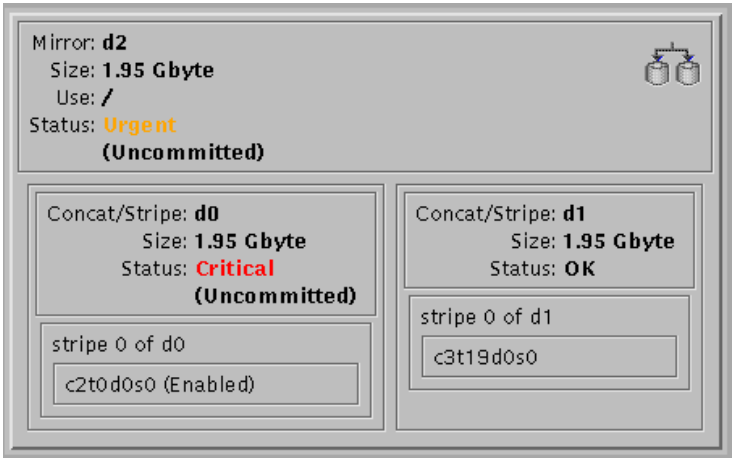


**FIGURE 1-30** Component Marked Critical Ready to be Committed

11. **To commit, click the `Mirror: d2`, and the Commit button located at the top of the canvas. The mirror will begin its resynchronization.**

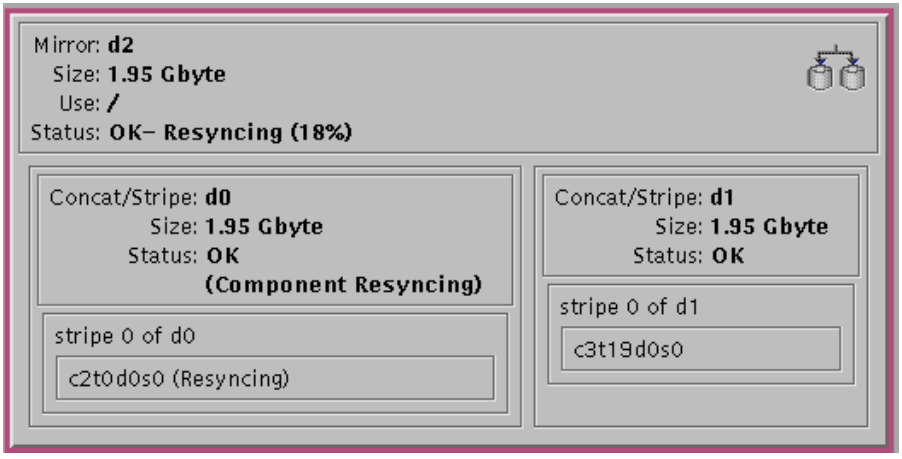


**FIGURE 1-31** Critical Component in Resynchronization State

12. **Repeat the same steps for all remaining Critical mirrored components, until the failed disk is repaired.**

    If you prefer to complete these steps through the command line, refer to Appendix A.

    You must reset the boot drive to the WWN of the newly inserted disk if you have `autoboot` set up. Currently, `luxadm` only accepts one boot device. If it is necessary to set up a secondary boot device, do it the next time the system is at the OpenBoot prompt. The current WWN is 22000020370c0c1c,0:a, which represents the failed, removed primary boot drive. The new primary boot drive WWN is w22000020370ca48f,0:a. CODE EXAMPLE 1-13 shows how to use the `luxadm` command to set up the new boot drive when the WWN has changed.

,

**CODE EXAMPLE 1-13** `luxadm` Command Sets Up the New Boot Drive

```
# eeprom boot-device
boot-device=/sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370c0c1c,0:a /sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370ca48f,0:a
# luxadm -v set_boot_dev /devices/sbus@1f,0/SUNW,socal@1,0/
sf@0,0/ssd@w22000020370ca48f,0:a
Current boot-device = /sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370c0c1c,0:a
 /sbus@1f,0/SUNW,socal@1,0/sf@0,0/ssd@w22000020370ca48f,0:a
New boot-device = /sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370ca48f,0:a
Do you want to change boot-device to the new setting? (y/n) y
# eeprom boot-device
boot-device=/sbus@1f,0/SUNW,socal@1,0/sf@0,0/
ssd@w22000020370ca48f,0:a
```

The system has now successfully been returned to a steady state with the primary boot disk mirrored to the secondary boot disk.

CHAPTER 2

# Mirroring with the Sun StorEdge A5000 Subsystem and the Sun Enterprise Volume Manager Software

This chapter guides you through the process of mirroring your boot/root/swap drive using the Sun Enterprise Volume Manager™ software (SEVM). The command-line utility with corresponding pictures of the GUI was chosen because completing this task with the GUI is too time-consuming and cumbersome. Starting with a new installation of SEVM, this chapter takes you from a failure of the primary boot disk to the recovery from that failure, and through the unencapsulation of the original primary boot disk. Ending in this manner helps ensure that a complete Solaris™ upgrade is possible. (Note that a patch installation, including a kernel patch or a volume manager patch, should be considered an upgrade.)

Some tips will be repeated at appropriate places throughout the chapter. Before and after encapsulating the root disk, it is a good idea to back up the `vtoc` to a file for future use, if necessary. You can do this by issuing the `prtvtoc /dev/rdsk/c#t#d#s2` command and piping to the proper file name on a disk other than the root disk. If you need to restore the `vtoc` in the future, do so by editing this file according to the `fmthard` manual page and supplying the file name to the `-s` option in the `fmthard` command. Checking the following files and commands ensures the system is operational along the way: `/etc/vfstab`, the mount and `df` command, the `prtvtoc` command to ensure that the two tags (14 and 15) for the public and private region are present.

The few rules that follow are required to encapsulate your root disk:

- Make sure that SEVM has room to work, by ensuring that you have two unassigned, zero-length slices available for the public and private region; historically, slices 3 and 4 are preferred. The amount of space the private region will consume can not be predetermined; however, the amount of space allocated for the private region will be between one and two cylinders, which is usually

between one and two Mbytes. A reasonable and safe precaution is to leave two cylinders unused and unassigned when partitioning your boot disk. Since we must align on cylinder boundaries, the space allocated is also dependent on the footprint of the disk. For extremely large configurations, it is possible that the private region requires more space. If the disk is already configured without any extra space, you might consider stealing two cylinders from the last used slice. These cylinders will not be set up as a slice; they will just be deallocated from the last slice. Re-label the disk, and verify that you can take this space without damaging the file system that resides on that slice. If the file system is near 90% full, you may not want to proceed in this direction. If you do not have two extra cylinders available, it will take what it needs from the swap space. If `swap` space is at a premium, other space should be allocated. Other caveats to allowing SEVM to steal from swap space include not having enough space left for a kernel core dump, which could result in the private region being overwritten. It is possible to dedicate a separate device for kernel core dumps. It is not advisable to allow the Volume Manager to control your swap space.

- The disk for encapsulation could contain `root`, `swap`, `usr`, `var`, or `opt`. Be aware that including other file systems such as `/home/export` will require moving that file system during unencapsulation.
- Using undocumented utilities is not recommended. For example, during the verification of this procedure, `vxprivutil` incorrectly reported that a newly encapsulated boot drive had a corrupted private region.

The documented configuration lists five controllers. Two of those controllers, c3 and c4, are attached to a dual loop A5000 subsystem. All other controllers were excluded from the configuration using the `/etc/vx/cntrls.exclude` file, which can be viewed in CODE EXAMPLE 2-1. All other disks have also been excluded for this specific example, and the file `/etc/vx/disk.exclude` is shown in CODE EXAMPLE 2-2.

**CODE EXAMPLE 2-1** `cat /etc/vx/cntrls.exclude`

```
c0
c1
c2
```

**CODE EXAMPLE 2-2** `cat /etc/vx/disks.exclude`

```
c3t35d0
c3t37d0
c3t50d0
c3t52d0
c4t3d0
c4t5d0
c4t18d0
```

# Configuring the SEVM Software

This section describes how to encapsulate and mirror the boot disk using the SEVM software.

## ▼ To Configure the SEVM Software

1. **Set up your path name to reflect the location of the SEVM software.**

**CODE EXAMPLE 2-3** Example of Volume Manager Path Name

```
# PATH=$PATH:/usr/sbin:/usr/bin:/opt/SUNWvxva/bin;export PATH
```

2. **Run** `vxinstall` **from command line.**

3. **Choose option 2, Volume Manager Custom Installation.**

This allows you to individually encapsulate a specified disk. You can also choose to initialize additional disks individually or ignore additional disks. In this example, we will encapsulate the boot disk and add a new disk to use as a mirror, as shown in Code Examples 2-3 through 2-8 and Figures 2-1 through 2-3.

**CODE EXAMPLE 2-4** Encapsulating the Boot Disk

```
Generating list of attached controllers....
(excluding c0>
(excluding c1>
(excluding c2>

Volume Manager Installation
Menu: VolumeManager/Install

  The Volume Manager names disks on your system using the controller
  and disk number of the disk, substituting them into the following
  pattern:

        c<controller>t<disk>d<disk>

  If the Multipathing driver is installed on the system then for the
  disk devices with multiple access paths, the controller number
  represents a multipath pseudo controller number. For example, if a
  disk has 2 paths from controllers c0 and c1, then the Volume
 Manager displays only one of them such as c0 to represent both the
  controllers.

  Some examples would be:

        c0t0d0  - first controller, first target, first disk
```

**CODE EXAMPLE 2-4** Encapsulating the Boot Disk *(Continued)*

```
        c1t0d0  - second controller, first target, first disk
        c1t1d0  - second controller, second target, first disk

  The Volume Manager has detected the following controllers on your
system:

        c3:
        c4:

Hit RETURN to continue.
Volume Manager Installation
Menu: VolumeManager/Install

  You will now be asked if you wish to use Quick Installation or
  Custom Installation.  Custom Installation allows you to select how
  the Volume Manager will handle the installation of each disk
  attached to your system.

  Quick Installation examines each disk attached to your system and
  attempts to create volumes to cover all disk partitions that might
  be used for file systems or for other similar purposes.

  If you do not wish to use some disks with the Volume Manager, or if
  you wish to reinitialize some disks, use the Custom Installation
  option Otherwise, we suggest that you use the Quick Installation
  option.
Hit RETURN to continue.
Volume Manager Installation Options
Menu: VolumeManager/Install

 1      Quick Installation
 2      Custom Installation

 ?      Display help about menu
 ??     Display help about the menuing system
 q      Exit from menus

Select an operation to perform: 2
Volume Manager Custom Installation
Menu: VolumeManager/Install/Custom

  The c4t1d0 disk is your Boot Disk.  You can not add it as a new
  disk.  If you encapsulate it, you will make your root filesystem
  and other system areas on the Boot Disk into volumes.  This is
  required if you wish to mirror your root filesystem or system
  swap area.

Encapsulate Boot Disk [y,n,q,?] (default: n) y

Enter disk name for c4t1d0 [<name>,q,?] (default: rootdisk)

  The c4t1d0 disk has been configured for encapsulation.

Hit RETURN to continue.
Volume Manager Custom Installation
Menu: VolumeManager/Install/Custom/c3
```

**CODE EXAMPLE 2-4** Encapsulating the Boot Disk *(Continued)*

```
Generating list of attached disks on c3....

<excluding c3t35d0>
<excluding c3t37d0>
<excluding c3t50d0>
<excluding c3t52d0>

  The Volume Manager has detected the following disks on controller
c3:

  c3t33d0

Hit RETURN to continue.

Installation options for controller c3
Menu: VolumeManager/Install/Custom/c3

 1      Install all disks as pre-existing disks. (encapsulate)
 2      Install all disks as new disks. (discards data on disks!)
 3      Install one disk at a time.
 4      Leave these disks alone.

 ?      Display help about menu
 ??     Display help about the menuing system
 q      Exit from menus

Select an operation to perform: 2

Volume Manager Custom Installation
Menu: VolumeManager/Install/Custom/c3/Init

Use default disk names for these disks? [y,n,q,?] (default: y) n

Enter disk name for c3t33d0 [<name>,q,?] (default: disk01) rootmir

Hit RETURN to continue.
Volume Manager Custom Installation
Menu: VolumeManager/Install/Custom/c4
Generating list of attached disks on c4....

<excluding c4t18d0>
<excluding root disk c4t1d0>
<excluding c4t3d0>
<excluding c4t5d0>
No disks were found attached to controller c4 !
Hit RETURN to continue.
Volume Manager Custom Installation
Menu: VolumeManager/Install/Custom

  The following is a summary of your choices.

        c3t33d0 New Disk
        c4t1d0  Encapsulate


Is this correct [y,n,q,?] (default: y) y
```

**CODE EXAMPLE 2-4** Encapsulating the Boot Disk *(Continued)*

```
# ls -l /dev/rdsk/c4t1d0s0
lrwxrwxrwx   1 root     root 74 Nov 11 12:44 /dev/rdsk/c4t1d0s0 ->
../../devices/sbus@3,0/SUNW,socal@0,0/sf@1,0/
ssd@w21000020370bfb2c,0:a,raw
# luxadm set_boot_dev /devices/sbus@3,0/SUNW,socal@0,0/sf@1,0/
ssd@w21000020370bfb2c,0:a
Do you want to change boot-device to the new setting? (y/n) y

# shutdown -g0 -y -i6
```

The system will reboot twice.

Note that at the end of the example above, the `luxadm` command is used to ensure that the proper boot device was defined so that the system will reboot using the correct disk.

**CODE EXAMPLE 2-5** Volume Manager Automatically Updates the `/etc/vfstab` File

```
#device              device                  mount   FS    fsck  mount   mount
#to mount            to fsck                 point   type  pass  at boot options
#
#/dev/dsk/c1d0s2     /dev/rdsk/c1d0s2        /usr    ufs   1     yes     -
fd                   -                       /dev/fd fd    -     no      -
/proc                -                       /proc   proc  -     no      -
/dev/vx/dsk/swapvol  -                       -       swap  -     no      -
/dev/vx/dsk/rootvol  /dev/vx/rdsk/rootvol    /       ufs   1     no      -
/dev/vx/dsk/usr      /dev/vx/rdsk/usr        /usr    ufs   1     no      -
/dev/vx/dsk/home     /dev/vx/rdsk/home       /home   ufs   2     yes     -
/dev/vx/dsk/opt      /dev/vx/rdsk/opt        /opt    ufs   2     yes     -
swap                 -                       /tmp    tmpfs -     yes     -
#NOTE: volume rootvol (/) encapsulated partition c3t1d0s0
#NOTE: volume swapvol (swap) encapsulated partition c3t1d0s1
#NOTE: volume opt (/opt) encapsulated partition c3t1d0s5
#NOTE: volume usr (/usr) encapsulated partition c3t1d0s6
#NOTE: volume home (/home) encapsulated partition c3t1d0s7
```

To see how SEVM has changed file system paths and how they are mounted, run `df` on the command line.

**CODE EXAMPLE 2-6** Mounted File Systems Shown with the `df` Command

```
# df
/                  (/dev/vx/dsk/rootvol): 5925804 blocks   480536 files
/usr               (/dev/vx/dsk/usr   ): 3179566 blocks   463320  files
/proc              (/proc             ):       0 blocks    15594  files
/dev/fd            (fd                ):       0 blocks        0  files
/home              (/dev/vx/dsk/home  ): 3809784 blocks   486590  files
/opt               (/dev/vx/dsk/opt   ):  891900 blocks   242610  files
/tmp               (swap              ): 3697712 blocks   165872  files
```

To see how SEVM has changed the system files, run `cat` <*filepath name*>, as shown in CODE EXAMPLE 2-7.

**CODE EXAMPLE 2-7** Lines Automatically Added to the `/etc/system` File

```
# cat /etc/system

     :
     :

* vxvm_START (do not remove)
forceload: drv/vxio
forceload: drv/vxspec
forceload: drv/ssd
forceload: drv/sf
forceload: drv/SUNW,socal
forceload: drv/sbus
forceload: drv/sd
forceload: drv/pln
forceload: drv/soc

#forceload: drv/socal
rootdev:/pseudo/vxio@0:0
set vxio:vol_rootdev_is_volume=1
* vxvm_END (do not remove)
```

---

**Note –** SEVM automatically sets up the `/etc/system` and `/etc/vfstab` files.

---

4. **Run** `vxva`.

   Note that you should have already run `vxinstall` so all the appropriate daemons are running. You can now continue the setup. SEVM will visually show you that the boot disk is encapsulated and is a member of `rootdg`.

   FIGURE 2-1 shows the GUI display of the encapsulated primary boot drive in the disk group `rootdg`. This encapsulation preserves the notion of the UNIX file system layout for future unencapsulation. However, when the root disk is mirrored, its mirror is a straight SEVM volume. Note that the mirror is unable to revert to a UNIX file system.

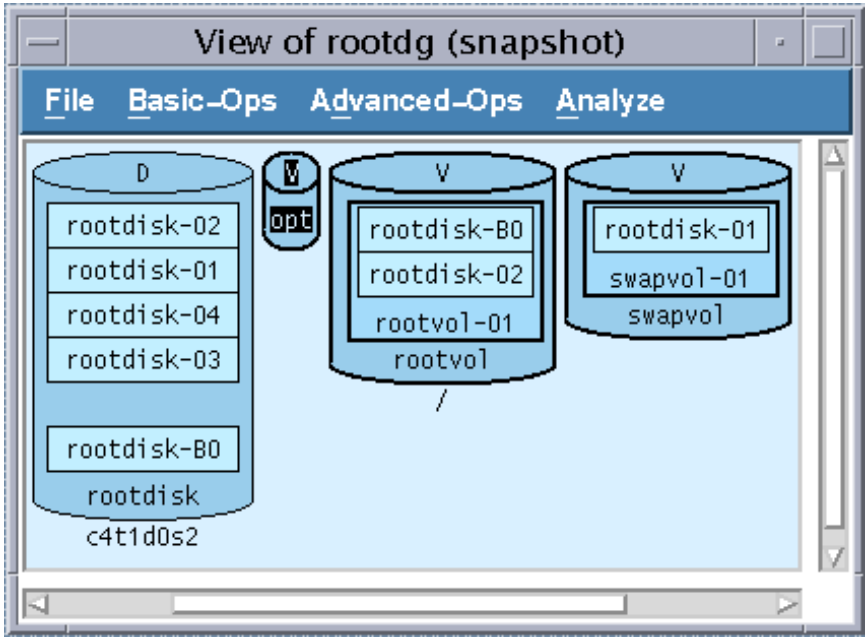


**FIGURE 2-1** GUI View of Encapsulated `rootdisk`

5. **Run** `vxdiskadm`**.**

   a. **Add the disk to the disk group.**

      Although this action was seemingly performed in Step 2, SEVM added the disk as a physical device but not as the disk named `rootmir`. So you must delete the disk and add it back. When adding the disk back, specify the name `rootmir`.

      FIGURE 2-2 shows that the mirror, disk c3t33d0s2 for the root disk, has now been properly named `rootmir`, instead of the device name c3t33d0s2. In the future, the name `rootmir` will make more sense and indicate that this is truly a mirror of the rootdisk.

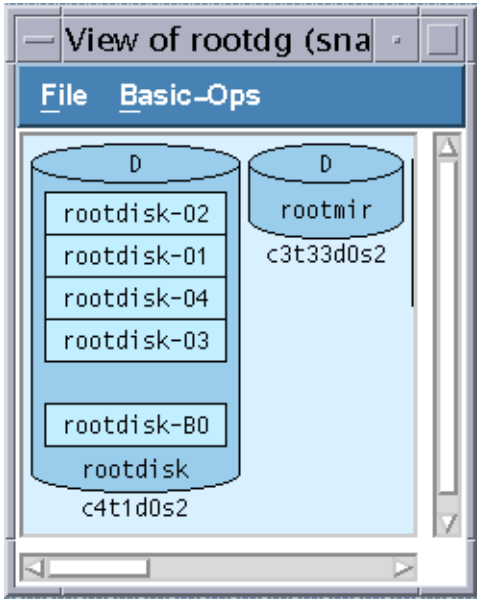


**FIGURE 2-2** GUI View of Spare Disk for Mirroring Root

**b. Mirror the disk, as shown in** CODE EXAMPLE 2-8**.**

Note the proper responses to the script used to mirror the data on the metadevice `rootdisk` to the metadevice `rootmir`.

**CODE EXAMPLE 2-8** `vxdiskadm` Command Used to Mirror Boot Disk

```
# vxdiskadm
Volume Manager Support Operations
Menu: VolumeManager/Disk

 1       Add or initialize one or more disks
 2       Encapsulate one or more disks
 3       Remove a disk
 4       Remove a disk for replacement
 5       Replace a failed or removed disk
 6       Mirror volumes on a disk
 7       Move volumes from a disk
 8       Enable access to (import) a disk group
 9       Remove access to (deport) a disk group
 10      Enable (online) a disk device
 11      Disable (offline) a disk device
 12      Mark a disk as a spare for a disk group
 13      Turn off the spare flag on a disk
 list    List disk information

 ?       Display help about menu
 ??      Display help about the menuing system
 q       Exit from menus
```

**CODE EXAMPLE 2-8** `vxdiskadm` Command Used to Mirror Boot Disk *(Continued)*

```
Select an operation to perform: 6
Mirror volumes on a disk

Menu: VolumeManager/Disk/Mirror

  This operation can be used to mirror volumes on a disk.  These
  volumes can be be mirrored onto another disk or onto any
  available disk space.  Volumes will not be mirrored if they are
  already mirrored.  Also, volumes that are comprised of more than
  one subdisk will not be mirrored.

  Mirroring volumes from the boot disk will produce a disk that
  can be used as an alternate boot disk.

  At the prompt below, supply the name of the disk containing the
  volumes to be mirrored.

Enter disk name [<disk>,list,q,?] rootdisk

  You can choose to mirror volumes from disk rootdisk onto any
  available disk space, or you can choose to mirror onto a specific
  disk.  To mirror to a specific disk, select the name of that disk.
  To mirror to any available disk space, select "any".

Enter destination disk [<disk>,list,q,?] (default: any) rootmir

  The requested operation is to mirror all volumes on disk rootdisk
  in disk group rootdg onto available disk space on disk rootmir.

  NOTE: This operation can take a long time to complete.

Continue with operation? [y,n,q,?] (default: y) y

  Mirror volume opt ...
  Mirror volume rootvol ...
  Mirror volume swapvol ...
  Mirror volume usr ...

  Mirroring of disk rootdisk is complete.

Mirror volumes on another disk? [y,n,q,?] (default: n) n
```

Veritas Visual Administrator (VxVA) displays the mirrored `rootdisk` as seen in FIGURE 2-3. The primary root disk is now fully mirrored. Keep track of the location of the primary disk. If you want to unencapsulate a disk for the purpose of performing a Solaris upgrade, it is required that the unencapsulation and the upgrade be performed on the primary disk, even if the primary disk has been replaced by a new disk because of a failure.

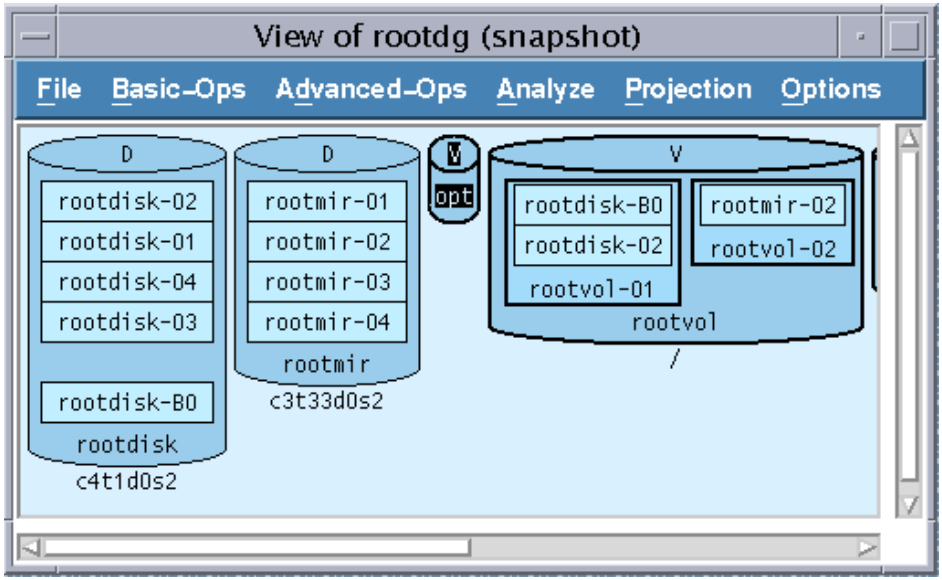


**FIGURE 2-3** GUI View of the Mirrored `rootdisk`

# Recovery

For an example of a failure, we have removed the primary boot device during a large file transfer. A corresponding real life scenario could be one in which someone accidentally pulls out an incorrect disk. The GUI (FIGURE 2-4) displays the failure to the primary boot disk in `rootdg`.

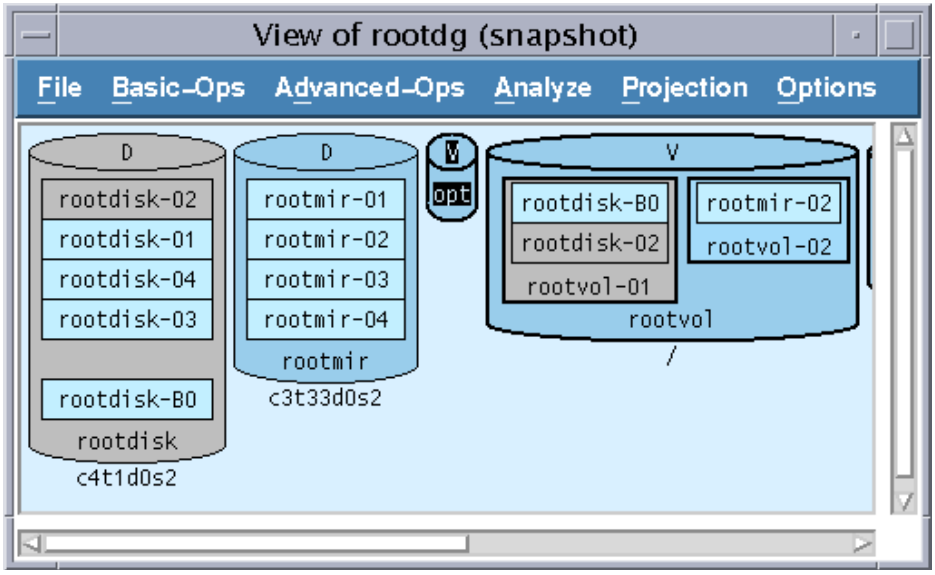


**FIGURE 2-4** Failure of Primary `boot` Disk

You can use the following command-line utilities to see a report of the primary disk failure. Code examples 2-9 through 2-11 show how each command reports the boot disk failure.

**CODE EXAMPLE 2-9** `vxdisk list` Command

```
# vxdisk list
DEVICE       TYPE      DISK         GROUP        STATUS
c3t33d0s2    sliced    disk01       rootdg       online
c3t35d0s2    sliced    -            -            online
c3t37d0s2    sliced    -            -            online
c3t50d0s2    sliced    -            -            online
c3t52d0s2    sliced    -            -            error
c4t1d0s2     sliced    -            -            online
c4t3d0s2     sliced    -            -            error
c4t5d0s2     sliced    -            -            online
c4t18d0s2    sliced    -            -            error
-            -         rootdisk     rootdg       failed failing was:c4t1d0s2
```

**CODE EXAMPLE 2-10** `vxdisk list c4t1d0s2` Command

```
# vxdisk list c4t1d0s2
vxvm:vxdisk: ERROR: Device c4t1d0s2: get_contents failed:
        Disk device is offline
Device:    c4t1d0s2
devicetag: c4t1d0
type:      sliced
```

**CODE EXAMPLE 2-10** vxdisk list c4t1d0s2 Command *(Continued)*

```
flags:     online error autoconfig
pubpaths:  block=/dev/vx/dmp/c4t1d0s3 char=/dev/vx/rdmp/c4t1d0s3
privpaths: block=/dev/vx/dmp/c4t1d0s4 char=/dev/vx/rdmp/c4t1d0s4
```

**CODE EXAMPLE 2-11** vxprint Command Output

```
# vxprint
Disk group: rootdg
TY_NAME          ASSOC       KSTATE      LENGTH    PLOFFS STATE     TUTIL0 PUTIL0
dg rootdg        rootdg      -           -         -      -         -      -

dm disk01        c3t33d0s2   -           17678493  -      -         -      -
dm rootdisk      -           -           -         -      NODEVICE  -      -

v home           fsgen       ENABLED     2869209   -      ACTIVE    -      -
pl home-02       home        ENABLED     2869209   -      ACTIVE    -      -
sd disk01-01     home-02     ENABLED     2869209   0      -         -      -
pl home-01       home        DISABLED    2869209   -      NODEVICE  -      -
sd rootdisk-03   home-01     DISABLED    2869209   0      NODEVICE  -      -

v opt            fsgen       ENABLED     2869209   -      ACTIVE    -      -
pl opt-02        opt         ENABLED     2869209   -      ACTIVE    -      -
sd disk01-02     opt-02      ENABLED     2869209   0      -         -      -
pl opt-01        opt         DISABLED    2869209   -      NODEVICE  -      -
sd rootdisk-05   opt-01      DISABLED    2869209   0      NODEVICE  -      -

v rootvol        root        ENABLED     6144201   -      ACTIVE    -      -
pl rootvol-02    rootvol     ENABLED     6144201   -      ACTIVE    -      -
sd disk01-03     rootvol-02  ENABLED     6144201   0      -         -      -
pl rootvol-01    rootvol     DISABLED    6144201   -      NODEVICE  -      -
sd rootdisk-B0   rootvol-01  DISABLED    1         0      NODEVICE  -      Block0
sd rootdisk-02   rootvol-01  DISABLED    6144200   1      NODEVICE  -      -

v swapvol        swap        ENABLED     2869209   -      ACTIVE    -      -
pl swapvol-02    swapvol     ENABLED     2869209   -      ACTIVE    -      -
sd disk01-04     swapvol-02  ENABLED     2869209   0      -         -      -
pl swapvol-01    swapvol     DISABLED    2869209   -      NODEVICE  -      -
sd rootdisk-01   swapvol-01  DISABLED    2869209   0      NODEVICE  -      -

v usr            fsgen       ENABLED     2869209   -      ACTIVE    -      -
pl usr-02        usr         ENABLED     2869209   -      ACTIVE    -      -
sd disk01-05     usr-02      ENABLED     2869209   0      -         -      -
pl usr-01        usr         DISABLED    2869209   -      NODEVICE  -      -
sd rootdisk-04   usr-01      DISABLED    2869209   0      NODEVICE
```

## ▼ To Recover the Primary Boot Disk

The following steps use `vxdiskadm` to lead you through recovery of the primary boot disk.

1. **Invoke `vxdiskadm` and choose option 4, Remove a disk for replacement, as shown in** CODE EXAMPLE 2-12**.**

**CODE EXAMPLE 2-12** `vxdiskadm` Command

```
# vxdiskadm
Volume Manager Support Operations
Menu: VolumeManager/Disk

 1      Add or initialize one or more disks
 2      Encapsulate one or more disks
 3      Remove a disk
 4      Remove a disk for replacement
 5      Replace a failed or removed disk
 6      Mirror volumes on a disk
 7      Move volumes from a disk
 8      Enable access to (import) a disk group
 9      Remove access to (deport) a disk group
 10     Enable (online) a disk device
 11     Disable (offline) a disk device
 12     Mark a disk as a spare for a disk group
 13     Turn off the spare flag on a disk
 list   List disk information

 ?      Display help about menu
 ??     Display help about the menuing system
 q      Exit from menus

Select an operation to perform: 4
Remove a disk for replacement
Menu: VolumeManager/Disk/RemoveForReplace

  Use this menu operation to remove a physical disk from a disk
  group, while retaining the disk name.  This changes the state
  for the disk name to a "removed" disk.  If there are any
  initialized disks that are not part of a disk group, you will be
  given the option of using one of these disks as a replacement.

Enter disk name [<disk>,list,q,?] rootdisk

  The following volumes will lose mirrors as a result of this
  operation:

        home opt rootvol swapvol usr

  No data on these volumes will be lost.

  The following devices are available as replacements:

        c3t35d0 c3t37d0 c3t50d0
```

**CODE EXAMPLE 2-12** `vxdiskadm` Command *(Continued)*

```
You can choose one of these disks now, to replace rootdisk.
  Select "none" if you do not wish to select a replacement disk.

Choose a device, or select "none"
[<device>,none,q,?] (default: c3t35d0) none

  The requested operation is to remove disk rootdisk from disk group
  rootdg.  The disk name will be kept, along with any volumes using
  the disk, allowing replacement of the disk.

  Select "Replace a failed or removed disk" from the main menu
  when you wish to replace the disk.

Continue with operation? [y,n,q,?] (default: y) y

  Removal of disk rootdisk completed successfully.

Remove another disk? [y,n,q,?] (default: n) n
```

2. **Use** `vxdisk` **to declare the disk device that has failed to be in the offline state, as shown in** CODE EXAMPLE 2-13**.**

**CODE EXAMPLE 2-13** `vxdisk` Command

```
# vxdisk offline c4t1d0
# vxdisk list
DEVICE       TYPE      DISK         GROUP        STATUS
c4t1d0s2     sliced    -            -            offline
```

You will be unable to proceed until the system recognizes that the primary encapsulated file systems are not in a busy state. The following error will occur if they are currently in a busy state: `Error: Could not offline device. May be Busy.`

3. **Use** `luxadm` **to remove the device, as shown in** CODE EXAMPLE 2-14**.**

**CODE EXAMPLE 2-14** `luxadm remove_device` Command

```
# luxadm remove_device snap0,f1
  WARNING!!! Please ensure that no filesystems are mounted on these device(s).
  All data on these devices should have been backed up.

The list of devices which will be removed is:
  1: Box Name "snap0" front slot 1

Please enter 'q' to Quit or <Return> to Continue:
```

**CODE EXAMPLE 2-14** luxadm remove_device Command

```
stopping:  Drive in "snap0" front slot 1....Done
offlining: Drive in "snap0" front slot 1....Done

Hit <Return> after removing the device(s).

  Drive in Box Name "snap0" front slot 1
    Removing Logical Nodes:
        Removing c4t1d0s0
        Removing c4t1d0s1
        Removing c4t1d0s2
        Removing c4t1d0s3
        Removing c4t1d0s4
        Removing c4t1d0s5
        Removing c4t1d0s6
        Removing c4t1d0s7
```

Although SEVM will correctly update its volume device path, you must update the logical device path with the correct WWN. This is especially important because this disk will be unencapsulated. So when the operating system is updated, this logical path name must be correct.

4. **Use** luxadm **to insert the new device to replace the primary boot disk, as shown in** CODE EXAMPLE 2-15**.**

**CODE EXAMPLE 2-15** luxadm insert_device Command

```
# luxadm insert_device snap0,f1
Notice: snap0,f1 may already be present.

The list of devices which will be inserted is:
  1: Box Name "snap0" front slot 1

Please enter ’q’ to Quit or <Return> to Continue:

Hit <Return> after inserting the device(s).
WARNING: /sbus@3,0/SUNW,socal@0,0/sf@1,0 (sf3): wwn changed on target 0x1

  Drive in Box Name "snap0" front slot 1

  Logical Nodes under /dev/dsk and /dev/rdsk :
        c4t1d0s0
        c4t1d0s1
        c4t1d0s2
        c4t1d0s3
        c4t1d0s4
```

**CODE EXAMPLE 2-15** `luxadm insert_device` Command *(Continued)*

```
        c4t1d0s5
        c4t1d0s6
        c4t1d0s7
```

5. **Invoke** `vxdiskadm` **and choose option 5, Replace a failed or removed disk, as shown in** CODE EXAMPLE 2-16**.**

**CODE EXAMPLE 2-16** `vxdiskadm` Option 5 - Replace a Failed or Removed Disk

```
Volume Manager Support Operations
Menu: VolumeManager/Disk

 1      Add or initialize one or more disks
 2      Encapsulate one or more disks
 3      Remove a disk
 4      Remove a disk for replacement
 5      Replace a failed or removed disk
 6      Mirror volumes on a disk
 7      Move volumes from a disk
 8      Enable access to (import) a disk group
 9      Remove access to (deport) a disk group
 10     Enable (online) a disk device
 11     Disable (offline) a disk device
 12     Mark a disk as a spare for a disk group
 13     Turn off the spare flag on a disk
 list   List disk information

 ?      Display help about menu
 ??     Display help about the menuing system
 q      Exit from menus

Select an operation to perform: 5

Replace a failed or removed disk
Menu: VolumeManager/Disk/ReplaceDisk

  Use this menu operation to specify a replacement disk for a disk
  that you removed with the "Remove a disk for replacement" menu
  operation, or that failed during use.  You will be prompted for
  a disk name to replace and a disk device to use as a replacement.
  You can choose an uninitialized disk, in which case the disk will
  be initialized, or you can choose a disk that you have already
  initialized using the Add or initialize a disk menu operation.

Select a removed or failed disk [<disk>,list,q,?] rootdisk

  The following devices are available as replacements:

        c3t35d0s2 c3t37d0s2 c3t50d0s2

  You can choose one of these disks to replace rootdisk.
Choose "none" to initialize another disk to replace rootdisk.

Choose a device, or select "none"
```

**CODE EXAMPLE 2-16** `vxdiskadm` Option 5 - Replace a Failed or Removed Disk *(Continued)*

```
[<device>,none,q,?] (default: c3t35d0s2) none

Select disk device to initialize [<address>,list,q,?] c4t1d0

  The following disk device appears to have been initialized already.
  The disk is listed as added to a disk group, but is not currently
  active.  Output format: [Device_Name,Disk_Access_Name,Disk_Group]

  [c4t1d0,c4t1d0s2,rootdg]

Use this device? [y,n,q,?] (default: y) y

  The following disk you selected for use appears to already have
  been initialized for the Volume Manager.  If you are certain the
  disk has already been initialized for the Volume Manager, then you
  do not need to reinitialize the disk device.
  Output format: [Device_Name]

  c4t1d0

Reinitialize this device? [y,n,q,?] (default: y) y

  The requested operation is to initialize disk device c4t1d0 and
  to then use that device to replace the removed or failed disk
  rootdisk in disk group rootdg.

Continue with operation? [y,n,q,?] (default: y) y

  Replacement of disk rootdisk in group rootdg with disk device
  c4t1d0 completed successfully.

Replace another disk? [y,n,q,?] (default: n) n
```

FIGURE 2-5 is a GUI snapshot of disks being synchronized.

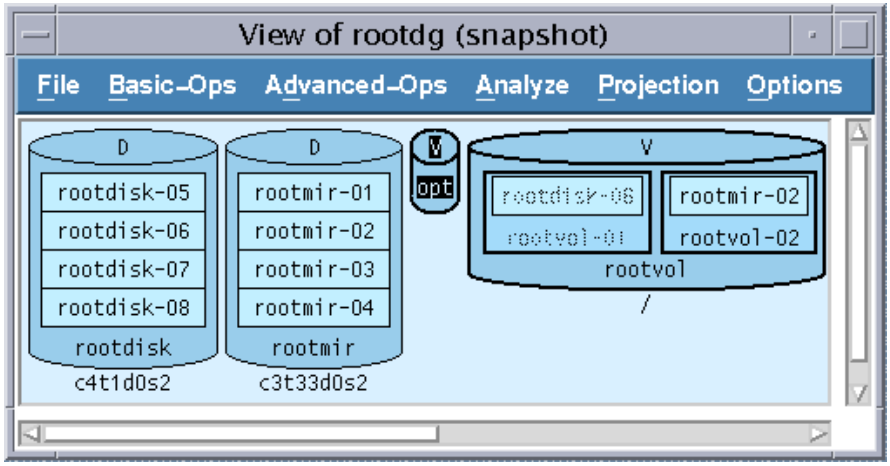


**FIGURE 2-5** Disks Being Synchronized

The replaced disk is now synchronized. Notice that the layout of the `rootdisk` is different from the original layout in FIGURE 2-1. Although the GUI layout appears different, this disk has knowledge of the Solaris file system and can be unencapsulated when an upgrade becomes necessary. It is important that you keep track of the original location of the encapsulated disk, so that only the volumes named `rootdisk` are attempted to be unencapsulated.

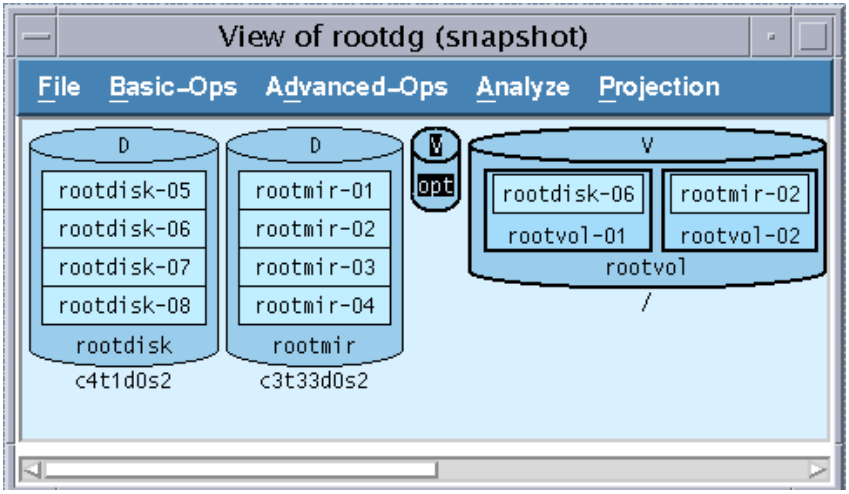


**FIGURE 2-6** The `rootdisk` Mirrored

6. **To complete the recovery, use the command-line interface to unmirror and unencapsulate the root device. You can use the GUI to disassociate and remove the mirror, but using the command-line interface is much more straightforward, simple, and quick.**

   **a. To unmirror the root device, use the commands shown in** CODE EXAMPLE 2-17**.**

**CODE EXAMPLE 2-17** Unmirroring the Root Device

```
# vxplex -o rm dis usr-02
# vxplex -o rm dis swapvol-02
# vxplex -o rm dis rootvol-02
# vxplex -o rm dis opt-02
```

FIGURE 2-7 shows the unmirrored `rootdisk`.

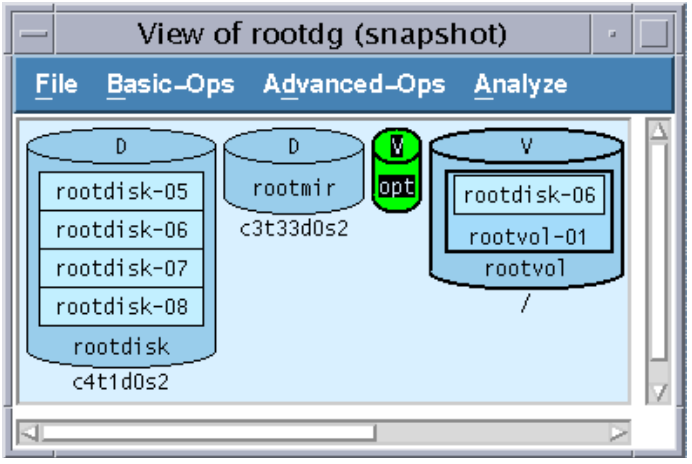


**FIGURE 2-7** The `rootdisk` Unmirrored

   **b. To unencapsulate the root device, run** `vxunroot`**.**

**CODE EXAMPLE 2-18** Unencapsulating `root`

```
# ./usr/lib/vxvm/bin/vxunroot

  This operation will convert the following file systems from
  volumes to regular partitions: root swap usr var opt

  Replace volume rootvol with c4t1d0s0.

  This operation will require a system reboot.  If you choose to
  continue with this operation, system configuration will be updated
  to discontinue use of the volume manager for your root and swap
  devices.

Do you wish to do this now [y,n,q,?] (default: y) y
```

**CODE EXAMPLE 2-18** Unencapsulating `root` *(Continued)*

```
  Restoring kernel configuration...

  A shutdown is now required to install the new kernel.
  You can choose to shutdown now, or you can shutdown later, at your
  convenience.

Do you wish to shutdown now [y,n,q,?] (default: n) y

  shutdown -g0 -y -i6

Shutdown started.    Fri Nov 13 17:09:21 EST 1998
```

**CODE EXAMPLE 2-19** Command-Line Tools Showing Results

```
# df
/                  (/dev/dsk/c4t1d0s0 ): 5820330 blocks   480739 files
/usr               (/dev/dsk/c4t1d0s6 ): 1994674 blocks   314140 files
/proc              (/proc             ):       0 blocks    15594 files
/dev/fd            (fd                ):       0 blocks        0 files
/opt               (/dev/dsk/c4t1d0s7 ): 2533110 blocks   336776 files
/tmp               (swap              ): 5841232 blocks   165846 files
/cdrom/sun_sevm_2_5_sparc(/vol/dev/dsk/c2t6d0/sun_sevm_2_5_sparc):       0
blocks        0 files
```

The updated `vfstab` file (see CODE EXAMPLE 2-20) shows that the devices have been updated to their original state and no longer are SEVM metadevices.

**CODE EXAMPLE 2-20** Results of `cat /etc/vfstab` Command

```
# cat /etc/vfstab
#device         device          mount           FS      fsck    mount   mount
#to mount       to fsck         point           type    pass    at boot options
#
#/dev/dsk/c1d0s2 /dev/rdsk/c1d0s2 /usr          ufs     1       yes     -
fd      -       /dev/fd fd      -       no      -
/proc   -       /proc   proc    -       no      -
/dev/dsk/c4t1d0s1       -       -       swap    -       no      -
/dev/dsk/c4t1d0s0       /dev/rdsk/c4t1d0s0      /       ufs     1       no      -
/dev/dsk/c4t1d0s6       /dev/rdsk/c4t1d0s6      /usr    ufs     1       no      -
/dev/dsk/c4t1d0s7       /dev/rdsk/c4t1d0s7      /opt    ufs     2       yes     -
swap    -       /tmp    tmpfs   -       yes     -

#NOTE: volume rootvol (/) encapsulated partition c4t1d0s0
#NOTE: volume swapvol (swap) encapsulated partition c4t1d0s1
#NOTE: volume opt (/opt) encapsulated partition c4t1d0s5
#NOTE: volume usr (/usr) encapsulated partition c4t1d0s6
```

CODE EXAMPLE 2-21 is yet another way you can verify that the devices have returned to their original state, which was unencapsulated.

**CODE EXAMPLE 2-21** Results of `mount` Command

```
# mount
/ on /dev/dsk/c4t1d0s0 read/write/setuid/largefiles on Fri Nov 13 17:12:44 1998
/usr on /dev/dsk/c4t1d0s6 read/write/setuid/largefiles on Fri Nov 13 17:12:44 1998
/proc on /proc read/write/setuid on Fri Nov 13 17:12:44 1998
/dev/fd on fd read/write/setuid on Fri Nov 13 17:12:44 1998
/opt on /dev/dsk/c4t1d0s7 setuid/read/write/largefiles on Fri Nov 13 17:13:13 1998
/tmp on swap read/write on Fri Nov 13 17:13:13 1998
```

# Adding Hot Relocation

A spare disk can be added to `rootdg` for online recovery of a failed primary boot drive. Note that this procedure is not recommended unless the group `rootdg` can have one and only one dedicated disk for this purpose.

A drawback to using a spare disk for this specific purpose is the inability to specify which disk the spare will recover. For example, given the scenario where you have mirrored the boot drive across two Sun StorEdge A5000 systems, as this chapter illustrates, you would add a spare to the Sun StorEdge A5000 system, which contains the secondary and/or mirrored drive. You could even place it on a separate loop, which would protect against everything but a power failure to that Sun StorEdge A5000 system. However, if the secondary disk were to fail, or another disk in the `rootdg` group failed, the spare disk would be used for recovery. In that case, the original plan of having a spare disk for your primary root disk may be thwarted. In addition, multiple volumes from separate disks may be placed on the spare, when ideally you want to keep your boot disk isolated.

SEVM has the ability to prevent a specific disk from being the target of data relocation, and also allows you to only use spare disks for data relocation as opposed to disks containing data. However, neither of these two scenarios allows you to have a one-to-one dedication linking a root disk to a specific spare.

CHAPTER 3

# Sun StorEdge MultiPack or UniPack as a Solution

This chapter addresses the need for a low-end solution for high availability. It provides the steps needed to configure a Sun StorEdge MultiPack or UniPack as the boot device. The examples here are based on the MultiPack or UniPack that contain the features of hot-swapability. The configuration included in this chapter was chosen to illustrate a previously undocumented scenario for boot/root/swap. In this chapter, the swap device is put on a dedicated disk, separate from the root file system, but included in the SEVM configuration. This configuration imposes specific constraints that you should be aware of before you proceed with it. Unless you are aware of these constraints, you may be unable to unencapsulate the root drive and upgrade the operating system.

## Hardware Configuration

Testing for this configuration was done with an Ultra 2 with two MultiPacks attached for redundancy. The same configuration using UniPack would be valid, stipulating that root and swap would reside on the same disk. In the recovery section, we refer to hot swapping the failed boot disk. Since the UniPack is a single disk in a doorless enclosure, its disk can not be swapped in or out. To replace the failed UniPack, power-off the failed disk enclosure, unplug the SCSI cable, replace the unit, and power it on. This UniPack must be alone or last on a SCSI chain.

# Software Configuration

The steps in this configuration are similar to those in Chapter 2. However, the command line interface is used for the initial setup of the encapsulation and mirroring. For the recovery portion of any encapsulated and mirrored disk using SEVM, you must use the scripts provided by SEVM. This chapter demonstrates using the `vxdiskadm` script for recovery. These scripts protect you from options and commands that should not be invoked, and allow for the recovery of the boot drive in several different scenarios.

One Gbyte is allocated for swap; the current memory size is 512 Mbyte. Although this amount of memory is overkill, systems running large applications may require 1 to 1.5 times that amount of RAM. The calculations are not discussed in this chapter, but appropriate sources can be found elsewhere. Unfortunately, there is no one-size-fits-all swap calculation.

TABLE 3-1 describes how the boot disk was partitioned and set up. The Solaris operating environment was installed on two separate disks, so that the configuration has an independent swap device. Disk `/dev/rdsk/c1t4d0`, contained `/`(root), `/opt`, and `/usr`. Disk `/dev/rdsk/c1t5d0` contained `swap`, and `/var`. This system was installed from the Solaris 2.6 Software (5/98) CD-ROM.

**TABLE 3-1** Disk layout

| File system / Mount point | Disk / Slice | Size |
|---|---|---|
| `/` | `c1t4d0s0` | `1800 MB` |
| `overlap` | `c1t4d0s2` | `4092 MB` |
| `/opt` | `c1t4d0s5` | `1000 MB` |
| `/usr` | `c1t4d0s6` | `1000 MB` |
| `swap` | `c1t5d0s1` | `1024 MB` |
| `overlap` | `c1t5d0s2` | `4092 MB` |
| `/var` | `c1t5d0s6` | `2800 MB` |

For the purpose of demonstration, swap is on a dedicated device. To ensure that SEVM has enough room for the encapsulation process, at least 1 to 2 cylinders, or approximately 1 to 2 MBytes of spare space have been left on each disk. Normally, when swap is contained on the same disk as the root file system, SEVM will steal

space, if needed, from swap space. But in this scenario, if the swap disk is entirely full and no free space is available, the following error will occur when the swap disk is encapsulated:

```
# /etc/vx/bin/vxencap -g rootdg swapdisk=c1t5d0
It is not possible to encapsulate c1t5d0, for the following reason:
     <vxvm:vxslicer: ERROR: Unsupported disk layout.>
```

---

**Tip –** For an explanation of the rules for encapsulating your boot drive, refer to Chapter 2, paragraph 3.

---

The configuration in this chapter uses a dedicated swap device. The appropriate time for configuring swap space onto a dedicated device is when installing the Solaris operating environment from a CD-ROM. During the installation process, you will be presented with a list of disk devices.

1. **From the list of devices, choose one disk for the root file system and any other file systems.**
2. **Choose a second device for the dedicated swap device.**

The system is now configured. The next step is the configuration of SEVM.

For a fresh installation of SEVM, covered in the previous chapter, run `vxinstall`. This chapter assumes that you have already installed SEVM. Therefore, a dummy disk is placed in the `rootdg` to simulate the existence of disks under SEVM control. The following steps show how to encapsulate and mirror the boot, root, and swap disk, using the command-line interface.

---

**Note –** SEVM uses a hot relocation algorithm such that any disk in the disk group with free space is a candidate to receive the relocated subdisks from the failed disk. In fact, even if no spare disk is specified for the diskgroup, subdisks will still get relocated to another disk provided that SEVM can find enough free space some place. Your root disk has the potential of being relocated anywhere in `rootdg` where there is free space at the time of failure, which is an undesirable outcome.

---

The following two ways to prevent this behavior:

1. Set the value spare=only in the `/etc/default/vxassist` file. The hot relocation algorithm (`vxrelocd`) uses only the disks defines as spare in that disk group and does not attempt to relocate the data to any other disk, even if free space is available. If not enough space is available on the spare disk(s), or if no spares are defined, no data relocation will occur.

2. Make every disk in `rootdg` ignored by hot relocation. Flag the disk as reserved with the `vxedit` command.

```
# vxedit set reserve=on <diskname>
```

This prevents SEVM from using the disk as a data relocation area. Of course, you will have to run this command on every disk in the diskgroup. A `vxdisk list` output will show the disk as being reserved.

**CODE EXAMPLE 3-1** `vxdisk list` command

```
# vxdisk list
c1t4d0s2     sliced     rootdisk     rootdg     online reserved
c1t5d0s2     sliced     swapdisk     rootdg     online reserved
c3t4d0s2     sliced     rootmir      rootdg     online reserved
c3t5d0s2     sliced     swapmir      rootdg     online reserved
```

This reservation flag prevents SEVM from using the disk for anything unless the disk is specified for use by a command, such as a `vxassist` or `vxmake` command.

## ▼ Command Line Configuration

1. **Encapsulate root and swap.**

```
# /etc/vx/bin/vxencap -g rootdg rootdisk=c1t4d0

  The c1t4d0 disk has been configured for encapsulation.
# /etc/vx/bin/vxencap -g rootdg swapdisk=c1t5d0

  The c1t5d0 disk has been configured for encapsulation.
# shutdown -g0 -y -i6
```

During the encapsulation phase, the system will reboot twice and you will see the following key messages on the console. The last two commands show how SEVM will configure the swap space.

**CODE EXAMPLE 3-2** Messages During Encapsulation Phase

```
VxVM general startup...

  The Volume Manager is now reconfiguring (partition phase)...

  Volume Manager: Partitioning c1t4d0 as an encapsulated disk.
```

**CODE EXAMPLE 3-2** Messages During Encapsulation Phase

```
VxVM general startup...

  Volume Manager: Partitioning c1t5d0 as an encapsulated disk.

  The Volume Manager is now reconfiguring (initialization phase)...

  Volume Manager: Adding rootdisk (c1t4d0) as an encapsulated disk.
Adding volumes for c1t4d0...
Starting new volumes...
Updating /etc/vfstab...
Remove encapsulated partitions...

The Volume Manager will now set up your Boot Disk as a managed disk.

Saving original configuration...
4 blocks

  Volume Manager: Adding swapdisk (c1t5d0) as an encapsulated disk.
Adding volumes for c1t5d0...
Starting new volumes...
Updating /etc/vfstab...
Remove encapsulated partitions...

  The system will now be rebooted.
syncing file systems... done
rebooting...
Resetting ...

# swap -l
swapfile             dev  swaplo blocks   free
/dev/vx/dsk/swapvol 137,7      16 2097344 2097344
# swap -s
total: 12864k bytes allocated + 4000k reserved = 16864k used,
1457424k available
```

Although we are using the command line to configure the boot disk, all changes will be reflected in the GUI, if a visual inspection is desired. FIGURE 3-1 shows us visually what the previous manual script has done. Root (/), /opt, /usr, and /var exist on /dev/rdsk/c1t4d0 (rootdisk), and the swap partition resides on /dev/rdsk/c1t5d0 (swapdisk).

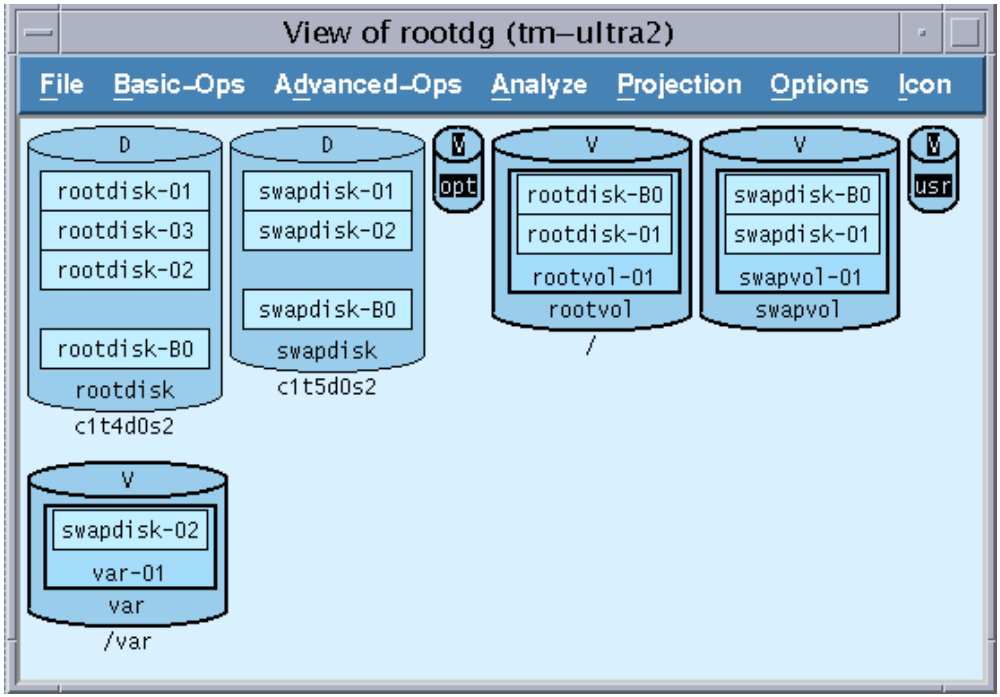


**FIGURE 3-1** GUI View of Encapsulated Boot Disk

The output in CODE EXAMPLE 3-3 from the `/etc/vfstab` and `df` command shows the changes that SEVM made to encapsulate root and `swap`.

**CODE EXAMPLE 3-3** Output from the `/etc/vfstab` and `df` Commands

```
# cat /etc/vfstab
#device         device          mount           FS      fsck    mount   mount
#to mount       to fsck         point           type    pass    at boot options
#
#/dev/dsk/c1d0s2 /dev/rdsk/c1d0s2 /usr          ufs     1       yes     -
fd      -       /dev/fd fd      -       no      -
/proc   -       /proc   proc    -       no      -
/dev/vx/dsk/swapvol     -       -       swap    -       no      -
/dev/vx/dsk/rootvol     /dev/vx/rdsk/rootvol    /       ufs     1       no      -
/dev/vx/dsk/usr /dev/vx/rdsk/usr        /usr    ufs     1       no      -
/dev/vx/dsk/var /dev/vx/rdsk/var        /var    ufs     1       no      -
/dev/vx/dsk/opt /dev/vx/rdsk/opt        /opt    ufs     2       yes     -
swap    -       /tmp    tmpfs   -       yes     -
#NOTE: volume rootvol (/) encapsulated partition c1t4d0s0
#NOTE: volume opt (/opt) encapsulated partition c1t4d0s5
#NOTE: volume usr (/usr) encapsulated partition c1t4d0s6
```

**CODE EXAMPLE 3-3** Output from the `/etc/vfstab` and `df` Commands *(Continued)*

```
#NOTE: volume swapvol (swap) encapsulated partition c1t5d0s1
#NOTE: volume var (/var) encapsulated partition c1t5d0s6
#
# df
/                  (/dev/vx/dsk/rootvol): 3524100 blocks   450387 files
/usr               (/dev/vx/dsk/usr   ): 1135724 blocks   462601 files
/proc              (/proc             ):       0 blocks     7946 files
/dev/fd            (fd                ):       0 blocks        0 files
/var               (/dev/vx/dsk/var   ): 5564702 blocks   466024 files
/opt               (/dev/vx/dsk/opt   ): 1668064 blocks   485173 files
/tmp               (swap              ): 2910704 blocks    84149 files
```

2. **Set up and enable the disk for the root and swap mirrors, using the `vxdisksetup` command, as shown in** CODE EXAMPLE 3-4**.**

**CODE EXAMPLE 3-4** `vxdisksetup` Command

```
# vxdisk list
DEVICE       TYPE      DISK         GROUP        STATUS
c3t3d0s2     sliced    -            -            error
c3t4d0s2     sliced    -            -            error

# /usr/lib/vxvm/bin/vxdisksetup -i c3t4d0
# /usr/lib/vxvm/bin/vxdisksetup -i c3t5d0

# vxdisk list
DEVICE       TYPE      DISK         GROUP        STATUS
c3t4d0s2     sliced    -            -            online
c3t5d0s2     sliced    -            -            online
```

FIGURE 3-2 is the visual result of bringing a new disk under SEVM control. Using `vxdisksetup` with the `-i` option simply creates a disk header on the disk and makes that disk directly usable by the volume manager. Previously, when only the disk on controller one and controller two were available, it was necessary to have the disk on controller three in our configuration since they are on our second Multipack unit.

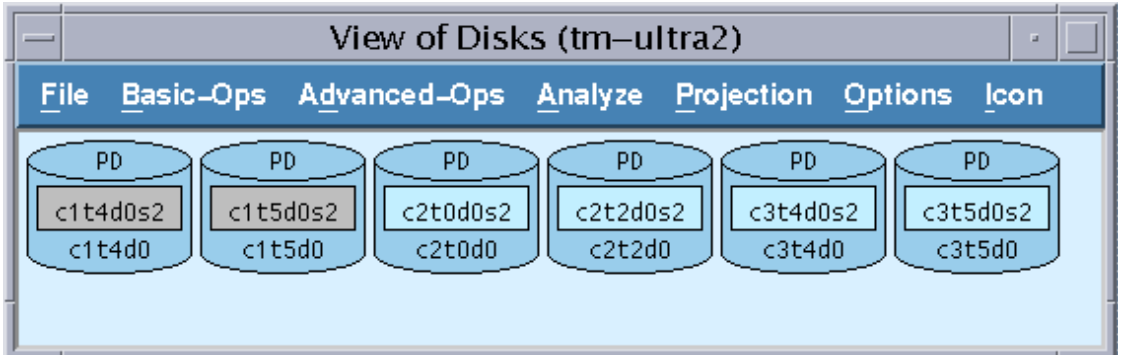


**FIGURE 3-2** GUI View of new disks brought under SEVM control

3. **Add disks to** `rootdg` **for the root and swap mirrors, as shown in** CODE EXAMPLE 3-5**.**

**CODE EXAMPLE 3-5** Adding Disks to `rootdg`

```
# /usr/sbin/vxdg -g rootdg adddisk rootmir=c3t4d0s2
# /usr/sbin/vxdg -g rootdg adddisk swapmir=c3t5d0s2

# vxdisk list
DEVICE       TYPE      DISK         GROUP        STATUS
c3t4d0s2     sliced    rootmir      rootdg       online
c3t5d0s2     sliced    swapmir      rootdg       online
```

Continuing with the visual inspection of our command line procedure, in FIGURE 3-3 we see that `rootmir`, and `swapmir` have been added successfully to the `rootdg` group. We now proceed to mirror our boot drive.

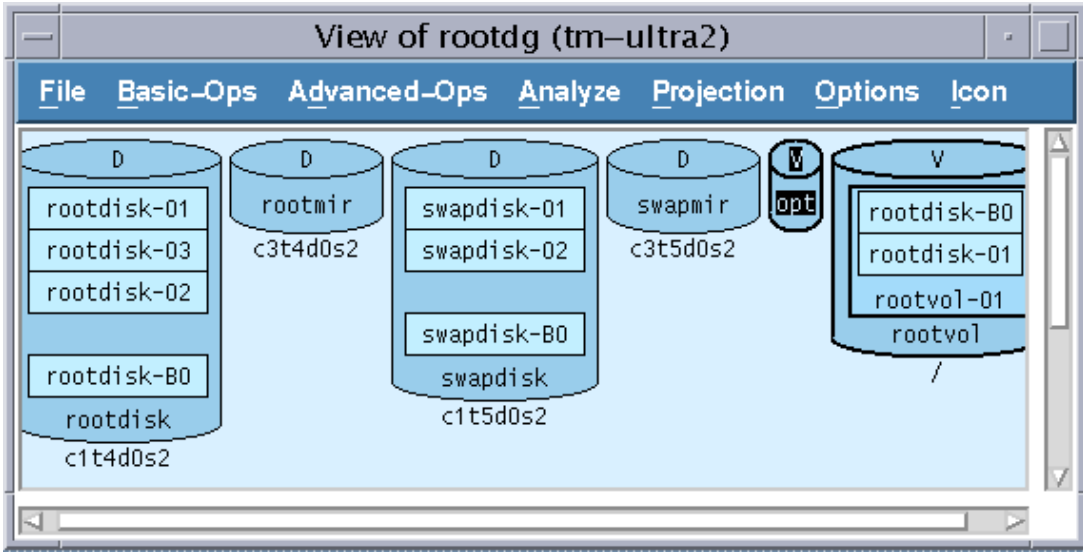


**FIGURE 3-3** GUI View of rootdg showing the results of the `addisk` command

4. **Mirror the root and swap disks, as shown in** CODE EXAMPLE 3-6**.**

**CODE EXAMPLE 3-6** `vxmirror` Command Used to Mirror the Root and Swap Disks

```
# /etc/vx/bin/vxmirror -g rootdg rootdisk rootmir
! vxassist -g rootdg mirror opt rootmir
! vxassist -g rootdg mirror rootvol rootmir
! vxassist -g rootdg mirror usr rootmir
! vxbootsetup


# /etc/vx/bin/vxmirror -g rootdg swapdisk swapmir
! vxassist -g rootdg mirror swapvol swapmir
! vxassist -g rootdg mirror var swapmir

# vxdisk list
DEVICE       TYPE      DISK         GROUP        STATUS
c1t4d0s2     sliced    rootdisk     rootdg       online
c1t5d0s2     sliced    swapdisk     rootdg       online
c3t4d0s2     sliced    rootmir      rootdg       online
c3t5d0s2     sliced    swapmir      rootdg       online
```

After the command `vxmirror` has completed, we can see in FIGURE 3-4 that our rootdisk and swapdisk have been successfully mirrored.

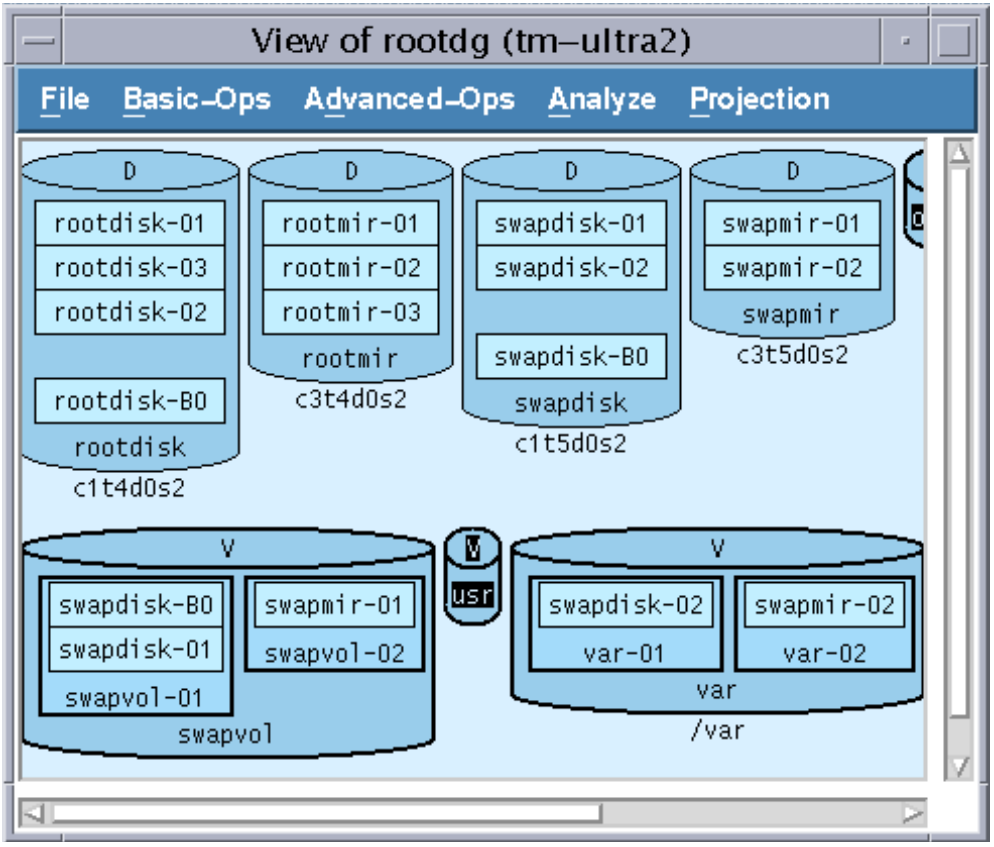


**FIGURE 3-4** GUI View of `vxmirror`

# Recovery

Use the `vxdiskadm` script for the recovery of a failed primary boot drive. SEVM ensures that proper steps are taken when replacing a disk. It makes a determination as to whether the disk being replaced is a boot disk or a regular SEVM disk. SEVM also determines the type of disk that is being used as a replacement. If care is not taken, the replaced boot disk could be formatted and configured incorrectly.

## ▼ To Recover the Failed Primary Boot Drive

1. **Invoke the** `vxdiskadm` **script.**

```
# vxdiskadm
```

2. **In the output generated, respond to the prompts in the script, as shown in** CODE EXAMPLE 3-7**.**

**CODE EXAMPLE 3-7** Output Generated from the `vxdiskadm` Script

```
Volume Manager Support Operations
Menu: Volume Manager/Disk

 1      Add or initialize one or more disks
 2      Encapsulate one or more disks
 3      Remove a disk
 4      Remove a disk for replacement
 5      Replace a failed or removed disk
 6      Mirror volumes on a disk
 7      Move volumes from a disk
 8      Enable access to (import) a disk group
 9      Remove access to (deport) a disk group
 10     Enable (online) a disk device
 11     Disable (offline) a disk device
 12     Mark a disk as a spare for a disk group
 13     Turn off the spare flag on a disk
 list   List disk information

 ?      Display help about menu
 ??     Display help about the menuing system
 q      Exit from menus

Select an operation to perform: 4
Remove a disk for replacement
Menu: VolumeManager/Disk/RemoveForReplace

  Use this menu operation to remove a physical disk from a disk
  group, while retaining the disk name.  This changes the state
  for the disk name to a "removed" disk.  If there are any
  initialized disks that are not part of a disk group, you will be
  given the option of using one of these disks as a replacement.

Enter disk name [<disk>,list,q,?] rootdisk

  The following volumes will lose mirrors as a result of this
```

**CODE EXAMPLE 3-7** Output Generated from the `vxdiskadm` Script

```
Volume Manager Support Operations
  operation:

        opt rootvol usr

  No data on these volumes will be lost.

  The following devices are available as replacements:

        c1t4d0 c2t0d0

  You can choose one of these disks now, to replace rootdisk.
  Select "none" if you do not wish to select a replacement disk.

Choose a device, or select "none"
[<device>,none,q,?] (default: c1t4d0) c1t4d0

Requested operation is to remove disk rootdisk from group rootdg.
  The removed disk will be replaced with disk device c1t4d0.

Continue with operation? [y,n,q,?] (default: y) y
  Removal of disk rootdisk completed successfully.

  Proceeding to replace rootdisk with device c1t4d0.

  Disk replacement completed successfully.

Remove another disk? [y,n,q,?] (default: n) n
```

# Upgrading the Solaris Operating Environment

To upgrade the Solaris operating environment after the root disk has been encapsulated with SEVM, you must unencapsulate the boot/root/swap disk. This section takes you though the procedure. After that, you will be ready to upgrade your system. If you have placed file systems other than `/`(root), `swap`, `/usr`, `/opt`, and `/var` on the boot drive, you will be asked to remove them before SEVM proceeds through the encapsulation. Because of this, you should put only the listed file systems onto the boot device.

## ▼ To Unencapsulate the boot/root/swap Disk

1. **Dissociate plexes and unmirror disks:**

```
# vxplex -o rm dis rootvol-02
# vxplex -o rm dis opt-02
# vxplex -o rm dis usr-02
# vxplex -o rm dis swapvol-02
# vxplex -o rm dis var-02
```

FIGURE 3-5 displays the results of Step 1. The plexes have been dissociated and removed in the same operation. Plex dissociation is the normal means of unmirroring, and makes the space used by those subdisks usable for new allocations. Our objective is simply to return to a Solaris root file system.

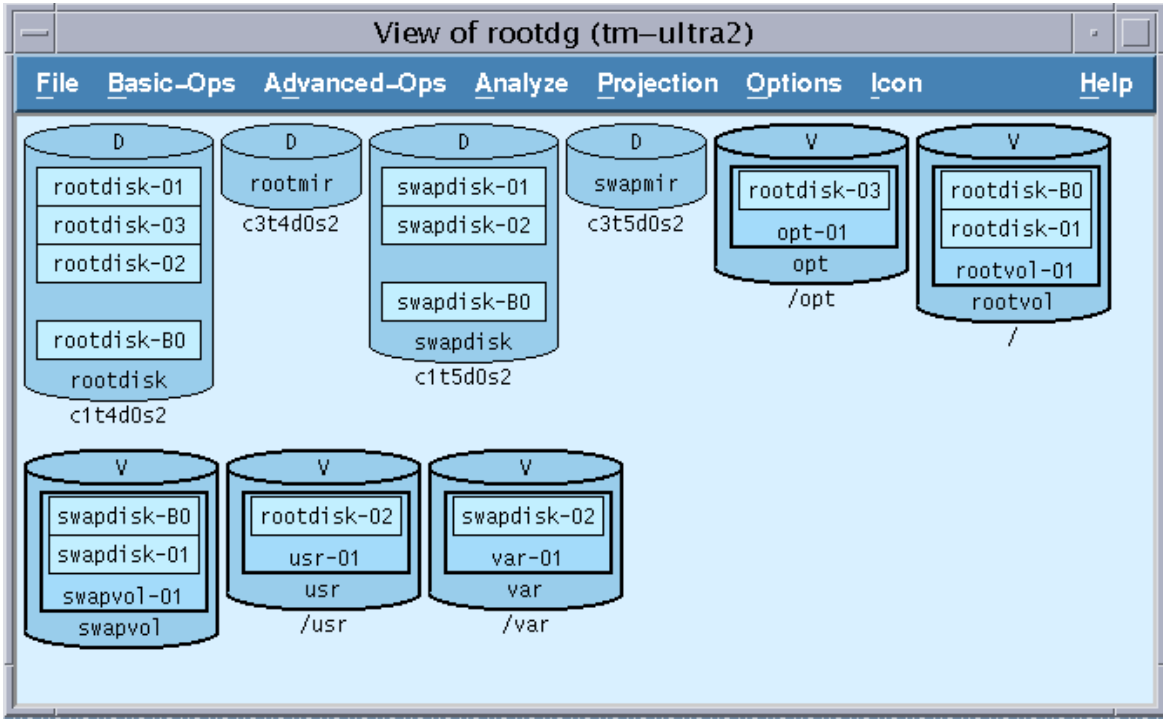


**FIGURE 3-5** GUI View of `vxplex`

2. **Run** `vxunroot`.

**CODE EXAMPLE 3-8** The `vxunroot` Script

```
# /usr/lib/vxvm/bin/vxunroot

  This operation will convert the following file systems from
  volumes to regular partitions: root swap usr var opt

  Replace volume rootvol with c1t4d0s0.

  This operation will require a system reboot.  If you choose to
  continue with this operation, system configuration will be
updated
  to discontinue use of the volume manager for your root and swap
  devices.

Do you wish to do this now [y,n,q,?] (default: y) y

  Restoring kernel configuration...

  A shutdown is now required to install the new kernel.
 You can choose to shutdown now, or you can shutdown later, at your
  convenience.

Do you wish to shutdown now [y,n,q,?] (default: n) y
```

While the machine reboots, you will get several messages from SEVM concerning the disk devices being removed from SEVM control. When you log in, you will find that the disks have been returned to their original Solaris state. This is depicted in CODE EXAMPLE 3-9. The last two commands display the swap space configuration.

**CODE EXAMPLE 3-9** Command-Line View of Disks Having Returned to Original State

```
# df
/                  (/dev/dsk/c1t4d0s0 ): 3524090 blocks   450404 files
/usr               (/dev/dsk/c1t4d0s6 ): 1135724 blocks   462601 files
/proc              (/proc             ):       0 blocks     7946 files
/dev/fd            (fd                ):       0 blocks        0 files
/var               (/dev/dsk/c1t5d0s6 ): 5564494 blocks   465991 files
/opt               (/dev/dsk/c1t4d0s7 ): 1668064 blocks   485173 files
/tmp               (swap              ): 2909968 blocks    84129 files

# cat /etc/vfstab
#device         device          mount           FS      fsck    mount   mount
#to mount       to fsck         point           type    pass    at boot options
#
#/dev/dsk/c1d0s2 /dev/rdsk/c1d0s2 /usr          ufs     1       yes     -
```

**CODE EXAMPLE 3-9** Command-Line View of Disks Having Returned to Original State

```
fd      -       /dev/fd fd      -       no      -
/proc   -       /proc   proc    -       no      -
/dev/dsk/c1t5d0s1       -       -       swap    -       no      -
/dev/dsk/c1t4d0s0       /dev/rdsk/c1t4d0s0      /       ufs     1       no
-
/dev/dsk/c1t4d0s6       /dev/rdsk/c1t4d0s6      /usr    ufs     1       no
-
/dev/dsk/c1t5d0s6       /dev/rdsk/c1t5d0s6      /var    ufs     1       no
-
/dev/dsk/c1t4d0s7       /dev/rdsk/c1t4d0s7      /opt    ufs     2       yes
-
swap    -       /tmp    tmpfs   -       yes     -
#NOTE: volume rootvol (/) encapsulated partition c1t4d0s0
#NOTE: volume opt (/opt) encapsulated partition c1t4d0s5
#NOTE: volume usr (/usr) encapsulated partition c1t4d0s6
#NOTE: volume swapvol (swap) encapsulated partition c1t5d0s1
#NOTE: volume var (/var) encapsulated partition c1t5d0s6
#
# swap -l
swapfile             dev  swaplo blocks   free
/dev/dsk/c1t5d0s1   32,161     16 2097344 2097344
# swap -s
total: 13192k bytes allocated + 3960k reserved = 17152k used, 1457056k available
#
```

CHAPTER 4

# Alternative Boot Solutions

Using a software disk manager is a popular solution for maintaining high availability for /boot/root/swap. However, alternatives exist that are commonly unknown or misconfigured. The configuration example in this chapter provides the steps that enable you to configure your boot disk to use Alternate Pathing (AP) in conjunction with SEVM. This solution is almost identical to the previous chapters on mirroring your boot drive, given that the mirrors exist on separate physical paths. The addition, AP software in this specific scenario adds the flexibility of being able to control which physical path from which you will access your boot drive. When a boot drive is under the mirrored configuration of Solstice DiskSuite (SDS) or SEVM, then the volume manager controls which path to the data is used. AP is a software layer which lies beneath the volume manager, giving you the ability to switch between primary and alternate paths to the data.

## Alternate Pathing and Sun Enterprise Volume Manager

AP software provides controller redundancy, and SEVM software provides data redundancy. If you plan to use AP software for making your boot disk highly available, you should also plan to use a compatible volume manager. The main purpose of AP software is to help protect against I/O controller failures. For disk controllers, this switch occurs automatically whenever a path failure is detected during normal operation.

## Overview of Alternate Pathing

In setting up the AP software, you set up a metadisk name and a metadisk pathgroup. The metadisk provides a means for you to access a disk, whereas the metadisk pathgroup provides a means for you to manipulate the path used to access that disk, such as in running AP commands to perform a switch from one path to another.

One of the alternate paths is the primary path which is initially the active alternate. The active alternate changes with a switch, but the primary path remains constant. You reference a disk pathgroup by specifying the `sf` port (for example, `sf1`) or for the SPARCstorage™ Array, the `pln` port (for example, `pln1`) that corresponds to the primary path. The command-line utility to switch paths is `apconfig`.

AP 2.1 software supports the Solaris 2.6 operating environment and Sun StorEdge A5000 subsystems attached to `sf` ports. AP software is compatible with SEVM 2.4 or later versions, but version 2.5 or later is preferred.

If you alternately path disks and also use a volume manager with those disks, the disks must be known to the volume manager by their AP software metadisk names. This requirement enables AP software to switch the active path without interfering with the volume manager.

You can place the boot disk interface under AP control. AP software makes it possible for the system to boot unattended even if the primary network or boot disk controller is not accessible, as long as viable alternate paths for these devices are defined.

# Hardware Configuration

The hardware configuration in this chapter was created using one Sun StorEdge A5000 system and two I/O controllers. This was used as a test platform to emulate a *portion* of the configuration seen in FIGURE 4-1. The configuration in FIGURE 4-1 is representative of *full* redundancy, and is the suggested configuration. With the configuration represented in FIGURE 4-1, we would be able to survive a power failure to one of the Sun StorEdge A5000 subsystems, a failed I/O board, or a failed component.

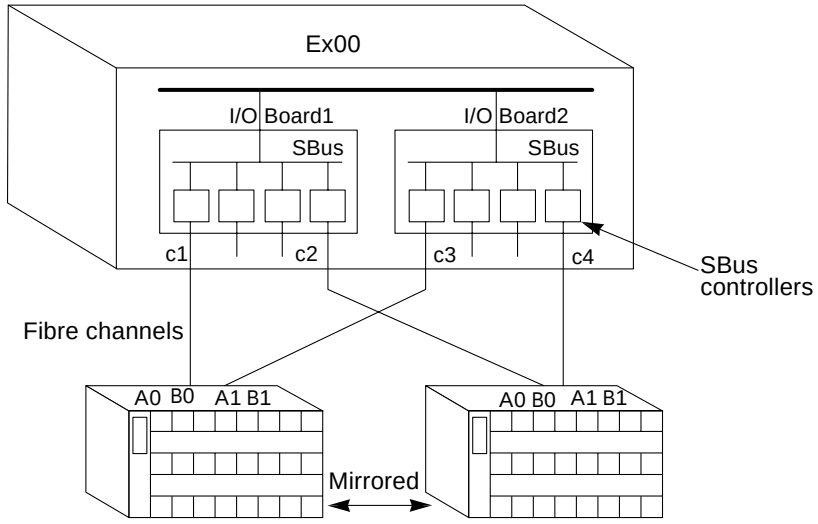


**FIGURE 4-1** Sun StorEdge A5000 System with One I/O Controller Board Mirrored

---

**Note –** FIGURE 4-1 shows the suggested configuration. When configuring a completely redundant system, the physical and metadisk names will differ from the example in this chapter. The following paragraph gives a detailed scenario.

---

In the sample configuration which follows, both the root (/), and its mirror are contained in the same A5000, in a single loop configuration. We thus obtain two physical paths to the data, creating a simulation environment. Note that the metadisk names for root (/) and its mirror *both* begin with controller two, mc2txd0. In FIGURE 4-1, we have root (/) located on the left A5000, and its mirror on the right A5000, giving four physical paths to two disks. In this case, if we chose the first disk in each enclosure, the physical paths for the left A5000 would be c1t32d0s0 (primary) and c3t32d0s0, and for the right c2t64d0s0 (primary), and c4t64d0s0. Therefore, the metadisk names for the primary root might start with controller one, mc1txd0, and the metadisk name for its mirror would start with a different controller, for example mc2txd0.

# Software Configuration

To create the example used in this chapter, Solaris 2.6 5/98 software and AP 2.1 from the Solaris 2.6 5/98 SMCC Supplement CD were installed. The required host packages for AP 2.1 for Sun Enterprise Servers, SUNWapdoc, SUNWapu, SUNWapr,

were also installed. Refer to the Sun Enterprise Server Release Notes for installation instructions. The `patch_cluster`, dated Feb/10/99, and the AP patch 106511-03 were also installed.

The system snapshot before installing AP or SEVM software is shown in CODE EXAMPLE 4-1 and CODE EXAMPLE 4-2.

**CODE EXAMPLE 4-1** System Snapshot Before AP and SEVM

```
# cat /etc/vfstab
#device             device               mount    FS     fsck  mount    mount
#to mount           to fsck              point    type   pass  at boot  options
#
#/dev/dsk/c1d0s2    /dev/rdsk/c1d0s2     /usr     ufs    1     yes      -
fd                  -                    /dev/fd  fd     -     no       -
/proc               -                    /proc    proc   -     no       -
/dev/dsk/c2t32d0s1  -                    -        swap   -     no       -
/dev/dsk/c2t32d0s0  /dev/rdsk/c2t32d0s0  /        ufs    1     no       -
/dev/dsk/c2t32d0s6  /dev/rdsk/c2t32d0s6  /usr     ufs    1     no       -
/dev/dsk/c2t32d0s5  /dev/rdsk/c2t32d0s5  /opt     ufs    2     yes      -
swap                -                    /tmp     tmpfs  -     yes
```

CODE EXAMPLE 4-2 displays that there are two paths to the data, on two separate controllers. The device names c2t32s0s0 and c1t32d0s0 both point to the same device. Although the second digit of the WWN differs, these are the same devices as seen through any disk format utility.

**CODE EXAMPLE 4-2** List of Devices

```
# ls -ltr /dev/dsk/c2t32d0s0
lrwxrwxrwx   1 root root 70 Mar 10 16:15 /dev/dsk/c2t32d0s0 ->
../../devices/sbus@6,0/SUNW,socal@1,0/sf@0,0/ssd@w2200002037079288,0:a
# ls -ltr /dev/dsk/c1t32d0s0
lrwxrwxrwx   1 root root 70 Mar 10 16:15 /dev/dsk/c1t32d0s0 ->
../../devices/sbus@3,0/SUNW,socal@0,0/sf@0,0/ssd@w2100002037079288,0:a
```

## ▼ To Configure Alternate Pathing

1. **Run the** `prtvtoc` **command to view the volume labels, as shown in** CODE EXAMPLE 4-3**.**

   Make sure there are no extra volumes that will be encapsulated.

**CODE EXAMPLE 4-3** Volume Label Displayed with `prtvtoc`

```
# prtvtoc /dev/rdsk/c2t32d0s2
* /dev/rdsk/c2t32d0s2 partition map
*
* Dimensions:
*     512 bytes/sector
*     133 sectors/track
*      27 tracks/cylinder
*    3591 sectors/cylinder
*    4926 cylinders
*    4924 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*
* Unallocated space:
*       First     Sector    Last
*       Sector     Count    Sector
*     4097331 4290869965 4294967295
*    17682084 4281382543   4097330
*     8194662   4097331  12291992
*    16389324   1289169  17678492
*
*                          First     Sector    Last
* Partition  Tag  Flags    Sector     Count    Sector  Mount Directory
       0      2    00          0   4097331   4097330
       1      3    01    4097331   4097331   8194661
       2      5    00          0  17682084  17682083
       3     14    01          0  17682084  17682083
       4     15    01   17678493      3591  17682083
       6      4    00   12291993   4097331  16389323
```

2. **Use the** `format` **command to allocate 300 Kbytes for AP databases.**

3. **Create the AP databases:**

```
# apdb -c /dev/rdsk/c2t33d0s4
```

**Note –** You must first allocate the space to hold the AP databases. If you do not first allocate this space, you will see the following error message and output from `apconfig -D`.

**CODE EXAMPLE 4-4** Errors Resulting From Not Allocating Space for AP Databases First

```
# apdb -c /dev/rdsk/c2t33d0s4
WARNING: ap_dbupdate: write error, errno = 5

WARNING: ap_sync_all_db: can't write database copy (118,52)
WARNING: ap_dbload: read error, errno = 5

path: /dev/rdsk/c2t33d0s4
major: 118
minor: 52
inaccessible: Yes


# apdb -d /dev/rdsk/c2t33d0s4
WARNING: ap_dbload: read error, errno = 5
```

With SEVM, two empty partitions are required, but allocation of the 1 to 2 cylinder space to be used in those empty partitions is automatically done by SEVM. You must have the 1 to 2 cylinders available for SEVM software to use; however, partitioning the equivalent 1 to 2 Mbytes is not required. The two products behave differently in the configuration of their internal tracking systems.

On some delivered configurations, this procedure is completed for you. If you are adding AP or SEVM software to your system, verify your particular layout and existing configuration.You must dedicate an entire disk partition, one that has at least 300 Kbytes, to each database copy. You should set up three to five database copies.

The database copies should have no I/O controllers or system boards in common with each other.

When deciding where to put the AP database, keep in mind that you will need to remove that database when unencapsulating the disk. (Note that the databases proceed through the encapsulation procedure and are represented with an

appropriate icon in vxvm.) To avoid this problem, find alternative locations for the AP database; do not put it on the disk to be encapsulated. This will also alleviate any partition shortage where SEVM and AP software combined require three partitions.

```
# apdb -c /dev/rdsk/c2t33d0s4 -f
# apdb -c /dev/rdsk/c1t33d0s4
# apdb -c /dev/rdsk/c2t48d0s4
# apdb -c /dev/rdsk/c1t48d0s4
```

In the example above, the `-c` option specifies the raw disk slice, and the `-f` (force) option is only necessary to create the first AP database copy.

If you want to place an AP database copy in a partition of an alternately pathed disk, create two copies of the database using each of the physical paths utilized by the AP metadisk to access the partition. Because the disk is accessible via two paths, AP software behaves as if two copies of the database exist, when actually, there is only one. The behavior does not result in database inconsistencies because AP software always updates and accesses database copies sequentially. This behavior does not result in performance problems because the AP database is not accessed frequently.

4. **Use** `apconfig` **to verify that you now have two copies of the AP database as shown in** CODE EXAMPLE 4-5**.**

**CODE EXAMPLE 4-5** AP Databases Displayed with `apconfig`

```
# apconfig -D

path: /dev/rdsk/c2t33d0s5
major: 118
minor: 12
timestamp: Mon Mar 11 08:38:24 1999
checksum: 3828031958
corrupt: No
inaccessible: No

path: /dev/rdsk/c1t33d0s4
major: 118
minor: 220
timestamp: Mon Mar 11 08:38:24 1999
checksum: 3828031958
corrupt: No
inaccessible: No

path: /dev/rdsk/c1t48d0s4
major: 118
minor: 212
timestamp: Mon Mar 11 08:38:24 1999
```

**CODE EXAMPLE 4-5** AP Databases Displayed with `apconfig` *(Continued)*

```
checksum: 3827963697
corrupt: No
inaccessible: No

path: /dev/rdsk/c2t48d0s4
major: 118
minor: 4
timestamp: Mon Mar 11 08:38:24 1999
checksum: 3827963697
corrupt: No
inaccessible: No
```

**Caution –** Do not create an AP database using a metadisk path (rather than a physical path to the disk slice.) If you use a metadisk path, the `apdb` command will succeed, but you may experience unpredictable behavior at a later time.

**Note –** AP software does not modify data when the disk is placed under AP control or when a pathgroup is deleted (except for AP database data). AP software does not repartition a disk.

5. **Determine which controllers are connected to the same devices in your system.**

   In CODE EXAMPLE 4-6, the `c` numbers are associated with the `sf` numbers. In this case, `sf10` and `sf6` point to the same devices, and they are attached to controllers `c1` and `c2`.

**CODE EXAMPLE 4-6** `apinst` Used to View of List of Attached Controllers

```
# apinst
sf10
        /dev/dsk/c2t48d0
        /dev/dsk/c2t33d0
        /dev/dsk/c2t49d0
        /dev/dsk/c2t34d0
        /dev/dsk/c2t54d0
        /dev/dsk/c2t53d0
        /dev/dsk/c2t50d0
        /dev/dsk/c2t35d0
        /dev/dsk/c2t51d0
        /dev/dsk/c2t32d0
        /dev/dsk/c2t36d0
        /dev/dsk/c2t37d0
        /dev/dsk/c2t52d0
        /dev/dsk/c2t38d0
isp0
        /dev/dsk/c3t5d0
sf6
        /dev/dsk/c1t49d0
        /dev/dsk/c1t34d0
        /dev/dsk/c1t53d0
        /dev/dsk/c1t54d0
        /dev/dsk/c1t50d0
        /dev/dsk/c1t35d0
        /dev/dsk/c1t51d0
        /dev/dsk/c1t32d0
        /dev/dsk/c1t36d0
        /dev/dsk/c1t37d0
        /dev/dsk/c1t52d0
        /dev/dsk/c1t38d0
        /dev/dsk/c1t48d0
        /dev/dsk/c1t33d0
```

6. **To alternately path the two `sf` controllers, create the metadisk with `apdisk`:**

```
# apdisk -c -p sf10 -a sf6
```

The primary path has been chosen to be `sf10`, since that corresponds to the physical

device path that you would normally boot on.

7. **Before committing the new metadisk in the AP database, verify that the uncommitted entry is correct by looking at the uncommitted SCSI disk entries.**

**CODE EXAMPLE 4-7** `apconfing -S -u` Command Used to View Uncommitted Entries

```
# apconfig -S -u
c1      sf6
c2      sf10  P A
        metadiskname(s):
                mc2t33d0  U
                mc2t48d0  U
                mc2t38d0  U
                mc2t52d0  U
                mc2t37d0  U
                mc2t36d0  U
                mc2t32d0  U
                mc2t51d0  U
                mc2t35d0  U
                mc2t50d0  U
                mc2t54d0  U
                mc2t53d0  U
                mc2t34d0  U
                mc2t49d0  U
```

8. **If everything looks fine, proceed with the commit, and then verify the database contents by looking at the committed SCSI entries.**

**CODE EXAMPLE 4-8** Verifying the Database Contents

```
# apdb -C
# apconfig -S
c1      sf6
c2      sf10  P A
        metadiskname(s):
                mc2t33d0
                mc2t48d0
                mc2t38d0
                mc2t52d0
                mc2t37d0
                mc2t36d0
                mc2t32d0
                mc2t51d0
                mc2t35d0
                mc2t50d0
                mc2t54d0
                mc2t53d0
                mc2t34d0
                mc2t49d0
```

The AP metadisk is built, but there are no devices corresponding with it in the Solaris device tree yet.

9. **Run the** `drvconfig` **command for the** `ap_dmd` **AP disk driver:**

```
# drvconfig -i ap_dmd
```

10. **Look at the devices that have been created:**

```
# ls -l /devices/pseudo/ap_dmd* | grep 72,blk
brw-rw-rw-   1 root     sys      136, 72 Mar 11 08:41 /devices/pseudo/ap_dmd@0:7
2,blk
# ls -l /devices/pseudo/ap_dmd* | grep 72,raw
crw-rw-rw-   1 root     sys      136, 72 Mar 11 08:41 /devices/pseudo/ap_dmd@0:7
2,raw
```

11. **Finally, create the appropriate links in** `/dev/ap/[r]dsk` **for the AP metadisks with the** `apconfig -R` **command:**

```
# apconfig -R
```

12. **Use the** `ls -ltr` **command to see that the appropriate links have been made:**

**CODE EXAMPLE 4-9** `ls-ltr` Command Used to Verify the New Links

```
# ls -ltr /dev/ap/dsk/* | grep mc2t32d0s0
lrwxrwxrwx   1 root     root          39 Mar 11 08:42 /dev/ap/dsk/mc2t32d0s0 ->
../../../devices/pseudo/ap_dmd@0:72,blk

#  ls -ltr /dev/ap/rdsk/* | grep mc2t32d0s0
lrwxrwxrwx   1 root     root          39 Mar 11 08:42 /dev/ap/rdsk/mc2t32d0s0 ->
 ../../../devices/pseudo/ap_dmd@0:72,raw
```

In CODE EXAMPLE 4-9, you again see the name of the device created in Step 9**.**

13. **Use** `apboot` **to define the new AP boot device.**

The `apboot` command modifies the `/etc/vfstab` and `/etc/system files`. In CODE EXAMPLE 4-10, `mc2t32d0` is the metadisk name of the boot disk. `apboot` examines the`/etc/vfstab` file and replaces the physical device name of the disk with the metadisk name. `apboot` also adds the kernel drivers to the system file, as shown in CODE EXAMPLE 4-10, so that they are loaded at boot time.

**CODE EXAMPLE 4-10** Replacing the Physical Device Name With the Metadisk Name

```
# apboot mc2t32d0
# apconfig -S

c1      sf6
c2      sf10  P A
        metadiskname(s):
                mc2t33d0
                mc2t48d0
                mc2t38d0
                mc2t52d0
                mc2t37d0
                mc2t36d0
                mc2t32d0  R
                mc2t51d0
                mc2t35d0
                mc2t50d0
                mc2t54d0
```

**CODE EXAMPLE 4-10** Replacing the Physical Device Name With the Metadisk Name

```
                mc2t53d0
                mc2t34d0
                mc2t49d0
```

CODE EXAMPLE 4-11 shows how AP software has changed the `vfstab` file path after executing the `apboot` command.

**CODE EXAMPLE 4-11** `/etc/vfstab` File Showing Metadisk Name

```
# cat /etc/vfstab
#device                device                  mount   FS    fsck mount   mount
#to mount              to fsck                 point   type  pass at boot options
#
#/dev/dsk/c1d0s2#      /dev/rdsk/c1d0s2        /usr    ufs   1    yes     -
fd                     -                       /dev/fd fd    -    no      -
/proc                  -                       /proc   proc  -    no      -
/dev/ap/dsk/mc2t32d0s1 -                       -       swap  -    no      -
/dev/ap/dsk/mc2t32d0s0 /dev/ap/rdsk/mc2t32d0s0 /       ufs   1    no      -
/dev/ap/dsk/mc2t32d0s6 /dev/ap/rdsk/mc2t32d0s6 /usr    ufs   1    no      -
/dev/ap/dsk/mc2t32d0s5 /dev/ap/rdsk/mc2t32d0s5 /opt    ufs   2    yes     -
swap                   -                       /tmp    tmpfs -    yes     -
```

CODE EXAMPLE 4-12 shows how AP software has changed the `/etc/system` file.

**CODE EXAMPLE 4-12** AP root Information Found in the `/etc/system` File

```
# cat /etc/system

     .
     .
     .

* Begin AP root info (do not edit)
forceload: drv/ssd
forceload: drv/sf
forceload: drv/SUNW,socal
forceload: drv/sbus
forceload: drv/ap_dmd
forceload: drv/ap
forceload: drv/pseudo
rootdev:/pseudo/ap_dmd@0:72,blk
* End AP root info (do not edit)
```

If you have any file systems that have metadevices but do not live on the boot disk, you must edit the `/etc/vfstab` file. For example, if your system has an independent boot device as described in Chapter 3, `/dev/rdsk/c2t2d0` would not be automatically replaced by a metadisk. You must edit the `/etc/vfstab` file and replace `/dev/[r]dsk/c2t2d0` with `/dev/ap/[r]dsk/mc2t2d0`.

14. **Set the OpenBoot PROM** `devalias` **variable boot-device to the physical path most likely to be used for booting** `luxadm`:

```
# eeprom boot-device
boot-device=/sbus@6,0/SUNW,socal@1,0/sf@0,0/ssd@w2200002037079288,0:a
```

15. **Define a** `devalias` **for the alternate boot device path as a convenience in case you have to perform a manual boot**:

```
luxadm set_boot_dev /sbus@6,0/SUNW,socal@1,0/sf@0,0/ssd@w2200002037079288,0:a
```

16. **Reboot the system**:

```
# shutdown -y -i6 -g0
```

17. **Install SEVM and patches, then configure SEVM to encapsulate the boot drive.**

---

**Note –** It may be simpler to put the controllers where you have created meta-devices for AP into the `/etc/vx/cntrls.exclude` file. This will cause SEVM to present only the metadevices and not the additional physical pathnames to those devices. Do this before installing SEVM. The file may look similar to CODE EXAMPLE 4-13.

---

**CODE EXAMPLE 4-13** `cntrls.exclude` File, Forcing Only AP Metadevices to be Displayed

```
# cat /etc/vx/cntrls.exclude
c1
c2
```

This allows you to mirror volumes as well as alternately path the I/O controllers to the disk arrays that make up those volumes.

SEVM 2.5 software contains dynamic multi-pathing (DMP), which provides an alternate pathing solution similar to AP software. AP software and DMP are not compatible and can not be used simultaneously within the same server or domain. You must decide which alternate pathing solution to use.

DMP supports automatic path switching for disks that are under control of SEVM. AP software supports disks that are under control of SEVM as well as disks that are not under control of SEVM. AP software also supports networks. AP 2.1 software supports automatic disk failover. AP 2.1 software does not, however, support automatic network failover, except at boot time.

If you install SEVM software while AP software is already configured, the SEVM installation script does not install DMP.

---

**Note –** I tested with SEVM patch 105463-06 and this patch broke AP. I backed up one patch, and patch 105463-05 worked for this example.

---

18. **Install Volume Manager using the** `vxinstall` **command.**

**CODE EXAMPLE 4-14** `vxinstall` Command used to Install Volume Manager

```
# vxinstall
Menu: VolumeManager/Install

  The Volume Manager names disks on your system using the controller
  and disk number of the disk, substituting them into the following
  pattern:

        c<controller>t<disk>d<disk>

  If the Multipathing driver is installed on the system then for the
  disk devices with multiple access paths, the controller number
  represents a multipath pseudo controller number. For example, if a
  disk has 2 paths from controllers c0 and c1, then the Volume Manager
  displays only one of them such as c0 to represent both the
  controllers.

  Some examples would be:

        c0t0d0  - first controller, first target, first disk
        c1t0d0  - second controller, first target, first disk
        c1t1d0  - second controller, second target, first disk

  The Volume Manager has detected the following controllers on your system:

        c0: sbus@3,0/SUNW,fas@3,8800000
        c3: sbus@7,0/QLGC,isp@0,10000
        mc2: pseudo

Hit RETURN to continue.
Volume Manager Installation
Menu: VolumeManager/Install
```

**CODE EXAMPLE 4-14** `vxinstall` Command used to Install Volume Manager

```
  You will now be asked if you wish to use Quick Installation or
  Custom Installation.  Custom Installation allows you to select how
  the Volume Manager will handle the installation of each disk
  attached to your system.

  Quick Installation examines each disk attached to your system and
  attempts to create volumes to cover all disk partitions that might
  be used for file systems or for other similar purposes.

  If you do not wish to use some disks with the Volume Manager, or if
  you wish to reinitialize some disks, use the Custom Installation
  option Otherwise, we suggest that you use the Quick Installation
  option.
Hit RETURN to continue.
```

**19. Choose Custom Installation:**

```
Volume Manager Installation Options
Menu: VolumeManager/Install

 1      Quick Installation
 2      Custom Installation

 ?      Display help about menu
 ??     Display help about the menuing system
 q      Exit from menus

Select an operation to perform: 2
```

**20. Choose to encapsulate the boot drive:**

```
Volume Manager Custom Installation
Menu: VolumeManager/Install/Custom

  The mc2t32d0 disk is your Boot Disk.  You can not add it as a new
  disk.  If you encapsulate it, you will make your root filesystem
  and other system areas on the Boot Disk into volumes.  This is
  required if you wish to mirror your root filesystem or system
  swap area.

Encapsulate Boot Disk [y,n,q,?] (default: n) y

Enter disk name for mc2t32d0 [<name>,q,?] (default: rootdisk)
```

```
Volume Manager Custom Installation
  The mc2t32d0 disk has been configured for encapsulation.

Hit RETURN to continue.
```

**21. Initialize drive for mirror.**

For this example, I encapsulated the boot drive and initialized the disk for the mirror, while ignoring all other drives.

```
Volume Manager Custom Installation
Menu: VolumeManager/Install/Custom/mc2
Generating list of attached disks on mc2....

<excluding root disk mc2t32d0>

  The Volume Manager has detected the following disks on controller mc2:

  mc2t33d0 mc2t34d0 mc2t35d0 mc2t36d0 mc2t37d0 mc2t38d0 mc2t48d0
  mc2t49d0 mc2t50d0 mc2t51d0 mc2t52d0 mc2t53d0 mc2t54d0

Hit RETURN to continue.
Installation options for controller mc2
Menu: VolumeManager/Install/Custom/mc2

 1      Install all disks as pre-existing disks. (encapsulate)
 2      Install all disks as new disks. (discards data on disks!)
 3      Install one disk at a time.
 4      Leave these disks alone.

 ?      Display help about menu
 ??     Display help about the menuing system
 q      Exit from menus

Select an operation to perform: 3
Installation options for disk mc2t33d0
Menu: VolumeManager/Install/Custom/mc2/mc2t33d0

 1      Install as a pre-existing disk. (encapsulate)
 2      Install as a new disk. (discards data on disk!)
 3      Leave this disk alone.

 ?      Display help about menu
 ??     Display help about the menuing system
 q      Exit from menus

Select an operation to perform: 3
```

```
Volume Manager Custom Installation
[ ... disks go by ... ]
Installation options for disk mc2t54d0
Menu: VolumeManager/Install/Custom/mc2/mc2t54d0

 1      Install as a pre-existing disk. (encapsulate)
 2      Install as a new disk. (discards data on disk!)
 3      Leave this disk alone.

 ?      Display help about menu
 ??     Display help about the menuing system
 q      Exit from menus

Select an operation to perform: 2

Are you sure (destroys data on mc2t54d0) [y,n,q,?] (default: n) y

Enter disk name for mc2t54d0 [<name>,q,?] (default: disk01) rootmir
Volume Manager Custom Installation
Menu: VolumeManager/Install/Custom

  The following is a summary of your choices.

        mc2t32d0        Encapsulate
        mc2t54d0        New Disk


Is this correct [y,n,q,?] (default: y) y
The system now must be shut down and rebooted in order to continue
the reconfiguration.

Shutdown and reboot now [y,n,q,?] (default: n) y

Shutdown started.    Thu Mar 11 12:06:00 PST 1999
```

**Note –** If you had an independent swap device, which would also be displayed as a metadisk, you would choose to encapsulate it also under the `vxinstall` menu item, Menu: VolumeManager / Install / Custom / mc2 / mc2txxd0. Reboot the system.

FIGURE 4-2 shows the GUI display of the root encapsulation we have just completed through `vxinstall`. In this snapshot, notice that the disk names are the metadisk names assigned by AP software.

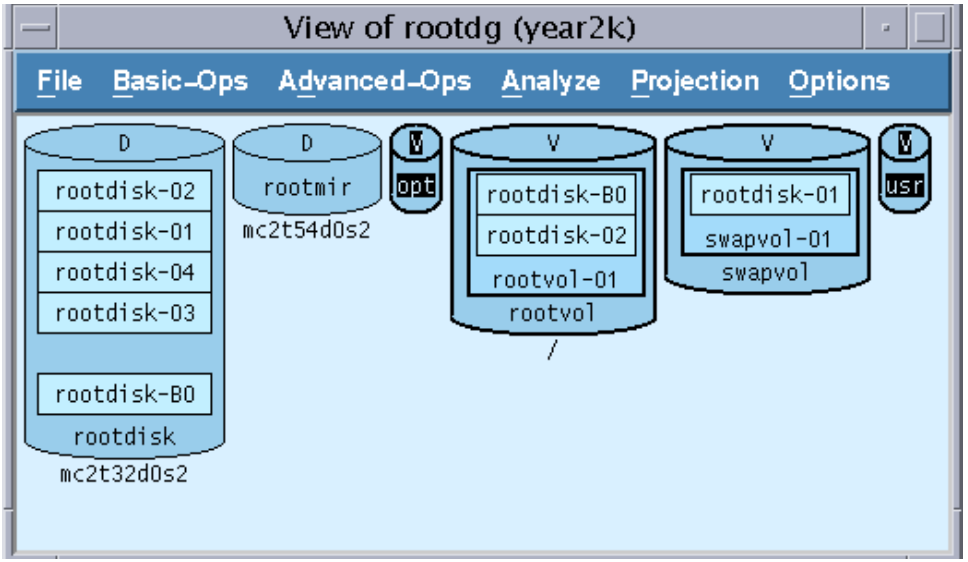


**FIGURE 4-2** GUI (VxVA) View of `rootdg`

## ▼ To Mirror the Boot Disk

1. **Notify AP about your boot disk mirror:**

```
# apboot -m mc2t54d0
# apconfig -S

c1      sf6
c2      sf10  P A
        metadiskname(s):
                mc2t33d0
                mc2t48d0
                mc2t38d0
                mc2t52d0
                mc2t37d0
                mc2t36d0
                mc2t32d0  R
                mc2t51d0
                mc2t35d0
                mc2t50d0
                mc2t54d0  M
```

```
                mc2t53d0
                mc2t34d0
                mc2t49d0
```

Initially, when running `vxinstall` you may or may not have added extra disks to your configuration for the purpose of mirroring your boot drive. If you did not add an extra disk while using `vxinstall`, you would be able to add the disk for the mirror by issuing the following commands after the machine reboots. In FIGURE 4-2, you can see that I have already performed step two since the `rootmir` object, mc2t54d0 already appears in the `rootdg` group.

2. **Set up and add disks to hold the mirrors:**

```
# /usr/lib/vxvm/bin/vxdisksetup -i mc2t54d0
# /usr/sbin/vxdg -g rootdg adddisk rootmir=mc2t54d0
```

3. **Create the mirrors of your boot disk through the use of the following SEVM command-line options:**

```
# /etc/vx/bin/vxmirror -g rootdg rootdisk rootmir
! vxassist -g rootdg mirror opt rootmir
! vxassist -g rootdg mirror rootvol rootmir
! vxassist -g rootdg mirror usr rootmir
! vxbootsetup
```

---

**Note –** Make sure that you have changed the OPB `boot-device` to the proper physical device from which you would normally choose to `boot`.You may choose to `boot` from any of the valid physical pathnames, which correspond to the boot device's AP meta-devices.

---

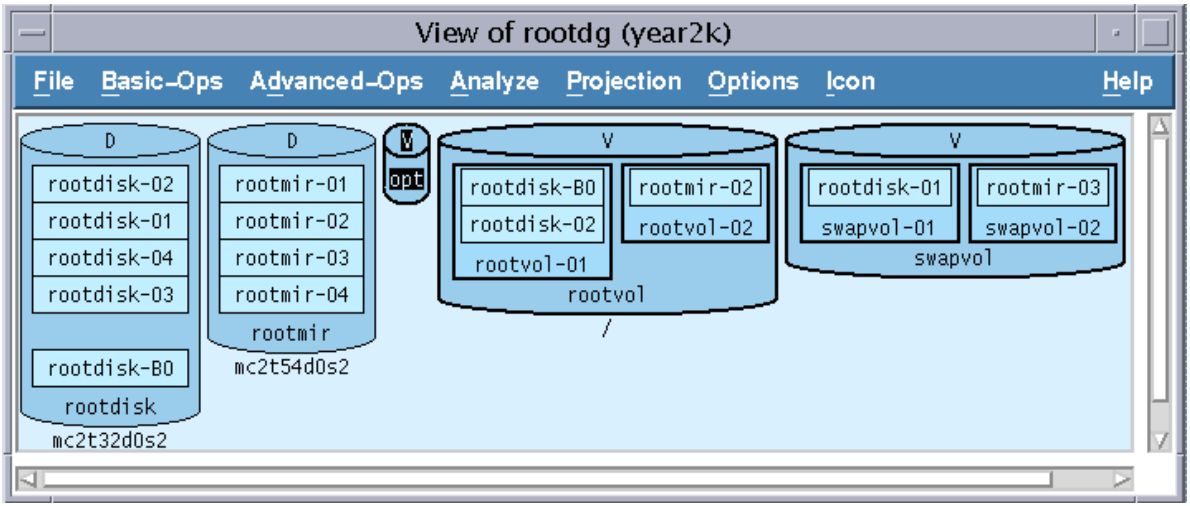


**FIGURE 4-3** Depiction of the Fully-Mirrored System Disk

The following diagram shows that the SPARCstorage Array (SSA) encapsulation would look the same as above, and also shows how the AP database would appear if it existed on the system boot disk.

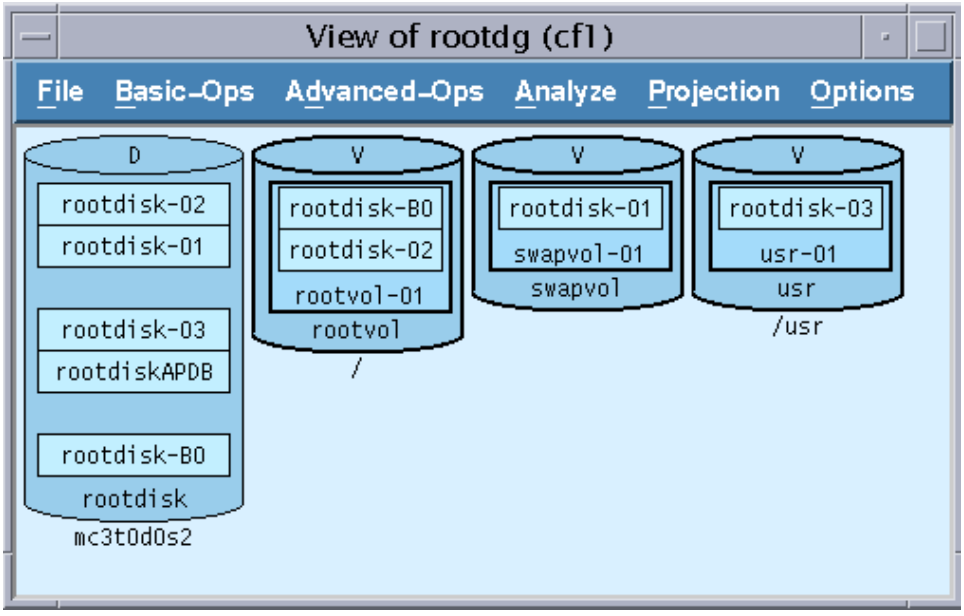


**FIGURE 4-4** System Snapshot Before AP and SEVM

APPENDIX **A**

# Using the Command Line

This appendix provides the command-line options that correspond to the GUI steps found in the text. The example in this appendix was performed on a system with two Sun StorEdge A5000 system disks that lay across power boundaries.

## Setup

1. **Create the initial state database replicas by typing the following commands sequentially:**

```
% metadb -a -f c0t1d0s3
% metadb -a -f c0t1d0s4
% metadb -a -f c1t18d0s3
% metadb -a -f c1t18d0s4
```

The `-a` and `-f` options are used together to create the initial replicas. The `-a` option attaches a new database device and automatically edits the `/etc/system` file. The `-f` creates the initial state database. These options are used together when no state databases exist.

2. **Create a mirror from root (/):**

```
% metainit -f d0 1 1 c0t1d0s0
```

Using the `-f` option forces the creation of the first concatenation (submirror1), even though the slice contains a mounted file system. The `d0` is then the name of the concatenation, and parameters `1 1` (numstripes width) represent the number of individual stripes on the metadevice, and the number of slices that make up a stripe. The last parameter is the component or logical name for the physical slice.

3. **Create the second concatenation (submirror2) for root (/):**

```
% metainit d1 1 1 c1t18d0s0
```

4. **Create a one-way mirror using the concatenation containing root (/). The parameters represent** `mirror -m` **submirror:**

```
% metainit d2 -m d0
```

5. **Edit the** `/etc/vfstab` **and** `/etc/system` **files:**

```
% metaroot d2
```

Running this command to edit the files also allows the system to be booted with the root file system (/) on a metadevice.

6. **Create a mirror for swap:**

```
% metainit -f d3 1 1 c0t1d0s1
```

7. **Create a second concatenation for swap:**

```
% metainit d4 1 1 c1t18d0s1
```

8. **Create a one-way mirror for swap:**

```
% metainit d5 -m d3
```

9. **Edit the** `vfstab` **file so that swap refers to the mirror you have just created (d5):**

```
% vi /etc/vfstab
```

10. **Create a mirror for** `/usr`**:**

```
% metainit -f d6 1 1 c0t1d0s6
```

11. **Create a second concatenation for** `/usr`**:**

```
% metainit d7 1 1 c1t18d0s6
```

12. **Create a one-way mirror for** `/usr`**:**

```
% metainit d8 -m d6
```

13. **Edit the** `vfstab` **file so that** `/usr` **refers to the mirror you have just created (d8):**

```
% vi /etc/vfstab
```

14. **Run the** `lockfs` **command:**

```
% lockfs -fa
```

15. **Reboot the system:**

```
% reboot
```

16. **After the reboot, attach the submirrors to the mirrors and cause a mirror resynchronization:**

```
% metattach d2 d1
% metattach d5 d4
% metattach d8 d7
```

17. **Run the `ls -l` command on the alternate root raw device to get the name of the alternate root device for boot information:**

```
% ls -l /dev/rdsk/c1t18d0s0
```

# Recovery

Use the following procedure when the primary boot disk becomes unreliable and needs to be replaced. When performing the recovery operation, be sure to recover the record of how your failed disk was formatted. Note that it should have been a mirror image of the boot drive that is still operational.

In the following steps, we will be hot-swapping the boot disk. Therefore, we will not reboot the system at any time. The procedure breaks all mirrors, removes the failed disk, inserts a new disk, prepares the disks, and reestablishes the mirrors that were broken.

1. **Delete all stale database replicas on the failed boot drive:**

```
% metadb -d c0t1d0s3
% metadb -d c0t1d0s4
```

2. **Detach the failed submirror root (/) from the mirror using the `-f` option (this forces the detach to occur):**

```
% metadetach -f d2 d0
```

3. **Clears the submirror using the `metaclear` command, which deletes all active configured metadevices:**

```
% metaclear d0
```

4. **Detach the failed submirror swap:**

```
% metadetach -f d5 d3
```

5. **Clear the active metadevice:**

```
% metaclear d3
```

6. **Detach the failed submirror** `/usr`:

```
% metadetach -f d8 d6
```

7. **Clear the active metadevice:**

```
% metaclear d6
```

8. **Use** `luxadm` **to hot remove the physical device from the A5000 subsystem:**

```
% luxadm remove /dev/rdsk/c0t1d0s2
```

This command checks if the device is busy, makes the device go offline, and informs the user that the device can be removed. It indicates which device can be removed by blinking the activity LED on the enclosure. In addition, it requests confirmation, and removes the logical device names for the device that was removed.

9. **Use** `luxadm` **to hot insert the new device:**

```
% luxadm insert A5000-1, f1
```

This command is interactive and informs the user when the device can be inserted, requests confirmation, creates new logical device names for the new devices, and displays the logical path name for those devices. From these new device names, you can determine the new WWN.

10. **Partition the disk to look exactly like the boot disk you previously removed:**

```
% format /dev/dsk/c0t1d0s2
```

11. **Add back the state database replica that you deleted in Step 1:**

```
% metadb -a c0t1d0s3
```

12. **Add back the state database replica that you deleted in Step 2:**

```
% metadb -a c0t1d0s4
```

13. **Recreate the submirror** `d0`**, root (/), with the new disk slice:**

```
% metainit d0 1 1 c0t1d0s0
```

14. **Attach the rebuilt submirror. A mirror resynchronization will occur automatically:**

```
% metattach d2 d0
```

15. **Recreate the submirror** `d3`**,** `swap`**, with the new disk slice:**

```
% metainit d3 1 1 c0t1d0s1
```

16. **Attach the rebuilt submirror. A mirror resynchronization will occur automatically.**

```
% metattach d5 d3
```

17. **Recreate the submirror** `d6`**,** `/usr`**, with the new disk slice:**

```
% metainit d6 1 1 c0t1ds6
```

18. **Attach the rebuilt submirror. A mirror resynchronization will occur automatically.**

```
% metattach d8 d6
```

19. **Set the boot device:**

```
% luxadm set_boot_device
```

# Index


## A

adding hot relocation
  with Sun StorEdge A5000 and SEVM, 56
Alternate Pathing
  configuring, 77
  overview, 74

## B

backup root, 19
building a MetaDB object, 12

## C

command line configuration
  with Sun StorEdge MultiPack or UniPack, 60
commands
  apboot, 84, 91
  apconfig, 79, 82 to 84, 91
  apdb, 78 to 79, 83
  apdisk, 81
  apinst, 81
  drvconfig, 83
  eeprom, 33, 86
  format, 6, 77, 99
  lockfs, 97
  luxadm
    inserting a device, 27, 50, 99
    removing a device, 26, 49, 99
    setting boot device, 28, 40, 100
  metaclear, 98 to 99
  metadb, 95, 98 to 100
  metadetach, 98 to 99
  metainit, 96 to 97
  metaroot, 96
  metastat, 23
  metatool, 9
  metattach, 97, 100
  mount, 56
  nvalias, 19
  prtvtoc, 77
  swap, 61, 71
  vxdg, 64, 92
  vxdisk list, 46, 60, 63
  vxdisk offline, 49
  vxdiskadm, 42 to 43, 48, 67
  vxdisksetup, 92
  vxedit, 60
  vxencap, 59 to 60
  vxinstall, 37, 87
  vxmirror, 65, 92
  vxplex, 54, 69
  vxprint, 47
  vxunroot, 54, 70
  vxva, 41
configuring Alternate Pathing, 77
configuring hardware
  with Sun StorEdge A5000 and SDS, 1
  with Sun StorEdge A5000 and two I/O controllers, 74
  with Sun StorEdge Multipack or Unipack, 57
configuring software
  with Solaris software and AP, 75
  with Sun StorEdge A5000 and SDS, 3

with Sun StorEdge A5000 and SEVM, 37
Creating AP databases, 77

## D

detecting a failed component
with Sun StorEdge A5000 and SDS, 20
with Sun StorEdge A5000 and SEVM, 46 to 47
DiskSuite Tool, starting, 9

## E

encapsulating boot disk, 38 to 40
with Sun StorEdge A5000 and SEVM, 37
encapsulating root disk, rules, 35

## H

hardware configuration
with Sun StorEdge A5000 and SDS, 1
with Sun StorEdge A5000 and two I/O controllers, 74
with Sun StorEdge MultiPack or UniPack, 57
hot relocation, adding, 56
hot-swapping boot disk using command line, 98

## I

initial state database replicas, 4

## M

MetaDBs, 5
metadisk driver, 3
mirroring boot disk
with Sun StorEdge A5000 and SEVM, 43
mirroring root disk
with Sun StorEdge A5000 and SDS, 13

## P

partitioning new disk
with Sun StorEdge A5000 and SDS, 28

## R

recovering a disk
using command line, 98
with Sun StorEdge A5000 and SDS, 24
with Sun StorEdge A5000 and SEVM, 48
with Sun StorEdge MultiPack or UniPack, 66
recovering disk using command line, 98
replacing primary disk
with SDS, 26
resynchronizing metadevice
with Sun StorEdge A5000 and SDS, 30
rules for encapsulating root disk, 35

## S

secondary root mirror, 19
setup using command line
with two Sun StorEdge 5000 system disks, 95
software configuration
with Solaris software and AP, 75
with Sun StorEdge MultiPack or UniPack, 58
Solaris software and AP
software configuration, 75
Solstice DiskSuite software
location, 4
majority consensus algorithm, 5
split-loop configuration, 3
state database replica, 4
creating, 6
definition, 5
space requirement, 5
two-disk configuration, 5
Sun StorEdge A5000 and SDS
building a MetaDB object, 12
creating initial state database replicas, 6
detecting a failed component, 20
hardware configuration, 1
mirroring root disk, 13
partitioning new disk, 28
recovering a disk, 24
software configuration, 3
synchronizing new disk, 28

Sun StorEdge A5000 and SEVM
adding hot relocation, 56
configuring software, 37
detecting a failed component, 46 to 47
encapsulating boot disk, 37
mirroring boot disk, 43
recovering a disk, 48
Sun StorEdge A5000 and two I/O controllers
hardware configuration, 74
Sun StorEdge A5000 Subsystem, 1
Sun StorEdge MultiPack or UniPack
command line configuration, 60
hardware configuration, 57
recovering a disk, 66
software configuration, 58
synchronizing new disk
with Sun StorEdge A5000 and SDS, 28

## T

two-disk configuration, 5

## U

Updating OpenBoot device parameter, 28
upgrading the Solaris operating environment, 68

## V

verifying database contents, 83
using, 69